We introduce a new class of numerical differentiation schemes constructed for the efficient solution of time-dependent PDEs that arise in wave phenomena. The schemes are constructed via the prolate spheroidal wave functions (PSWFs). Compared to existing differentiation schemes based on orthogonal polynomials, the new class of differentiation schemes requires fewer points per wavelength to achieve the same accuracy when it is used to approximate derivatives of bandlimited functions. In addition, the resulting differentiation matrices have spectral radii that grow asymptotically as m for the case of first derivatives, and m^2 for second derivatives, with m being the dimensions of the matrices. The above results mean that the new class of differentiation schemes is more efficient in the solution of time-dependent PDEs compared to existing schemes such as the Chebyshev collocation method. The improvements are particularly prominent in large-scale time-dependent PDEs, in which the solutions contain large numbers of wavelengths in the computational domains.

A new class of highly accurate differentiation schemes based on the prolate spheroidal wave functions

W. Y. Kong [†] and V. Rokhlin [*]

Technical Report YaleU/DCS/TR-1444
April 7, 2011

[†] Dept. of Mathematics, Yale University, New Haven CT 06511
[*] Dept. of Mathematics and Computer Science, Yale University, New Haven CT 06511

The authors were supported in part by the Office of Naval Research under contract N00014-07-1-0711 and the Air Force Office of Scientific Research under contract FA9550-09-1-0241.

Approved for public release: distribution is unlimited.
Keywords: *Prolate spheroidal wave functions, numerical differentiation*

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
07 APR 2011		**00-00-2011 to 00-00-2011**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A new class of highly accurate di erentiation schemes based on the prolate spheroidal wave functions	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Yale University ,Department of Computer Science,New Haven,CT,06520**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**72**	

1 Introduction

The numerical solution of partial differential equations (PDEs) is ubiquitous in scientific computations, and as such it is a well-developed subject that has been widely studied (see, for example, [1, 21, 34, 53, 62]). One important class of PDEs is the time-dependent PDEs, which are PDEs that involve derivatives in both time and spatial dimensions. This kind of PDEs arises in the modeling of physical phenomena in many scientific disciplines, such as thermodynamics, electromagnetics, and fluid mechanics. Usually, the values or the derivatives of the solution at time $t = 0$ (the initial conditions) and on the boundary of the spatial domain (the boundary conditions) are specified in order to guarantee uniqueness of the solution. A common approach to the solution of a time-dependent PDE is to first approximate the spatial derivatives by a numerical differentiation scheme, discretizing the spatial derivative operators. This converts the PDE into a linear system of ordinary differential equations (ODEs) in time. The system is then solved by a numerical ODE solver, such as a Runga-Kutta scheme or a predictor-corrector scheme. Such an approach for the solution of time-dependent PDEs, which is sometimes referred to as the "method of lines," is studied in [54, 55], and is applied in a wide range of context (see, for instance, [39, 41, 61, 63]).

The above approach for the solution of time-dependent PDEs requires the discretization of the spatial derivatives, for which many numerical differentiation methods are available. A classical method, which dates back to the first half of the nineteenth century, is the finite difference method. It approximates the derivative of a function $f : \mathbb{R} \to \mathbb{R}$ at a point x by constructing a polynomial p that interpolates f at some set of equispaced points around x, and then taking the derivative of the polynomial p at x. This method, although simple to implement, has low orders of accuracy, and is not suitable for the solution of large-scale PDEs.

The last few decades saw a major advance in the class of *spectral methods* for numerical differentiation and the numerical solution of PDEs. It was pioneered by the work of Gottlieb, Orszag, and others, who demonstrated that spectral methods are particularly applicable to problems in fluid dynamics ([11, 27, 30]). Subsequent development of spectral methods has been contributed by the work of Mercier [51], Funaro [24], Fornberg [23], Trefethen [63], and Boyd [8].

The main idea of spectral methods is to approximate a function f by a finite series u_n of smooth functions $\phi_1, \ldots, \phi_n$, and then approximate the derivatives of f by the derivatives of the series u_n. In the context of a time-dependent PDE, it involves approximating the solution u of the PDE by the finite series

$$u(x,t) \approx u_n(x,t) = \sum_{k=1}^{n} a_k(t)\phi_k(x), \tag{1.1}$$

where the coefficients $a_1, \ldots, a_n$ and the functions $\phi_1, \ldots, \phi_n$ are chosen according to the particular type of spectral methods employed. Among these types of spectral methods are *tau*, *Galerkin*, and *collocation* methods. The tau and the Galerkin methods involve converting the time-dependent PDE, together with its initial and boundary conditions, into a system of ODEs satisfied by the coefficients $a_1, \ldots, a_n$. The solution for $a_1, \ldots, a_n$ then provides a solution for u. When the coefficients $a_1, \ldots, a_n$ and the functions

$\phi_1, \ldots, \phi_n$ are suitably chosen, these methods attain *spectral accuracy*; namely, the rates of convergence of their accuracies depend only on the smoothness of the solution u (see, for example, [30]). In particular, when the solution u is analytic, the errors decay exponentially with the number of functions n. Common choices of $\phi_1, \ldots, \phi_n$ are trigonometric functions for periodic problems, and Chebyshev or Legendre polynomials for non-periodic problems.

The collocation method, which is also referred to as the *pseudospectral method* (see [30]), takes a slightly different approach. First, a set of points $x_1, \ldots, x_m$ in the spatial domain, called the collocation points, are chosen. Then, the coefficients $a_1, \ldots, a_n$ are computed such that the series u_n in (1.1) exactly equals u at the points $x_1, \ldots, x_m$, and that u_n satisfies the boundary conditions. This interpolatory procedure gives rise to an $m \times m$ *differentiation matrix* D that takes the values of u at the collocation points $x_1, \ldots, x_m$ to the approximate values of the spatial derivatives of u at $x_1, \ldots, x_m$. The "method of lines" as described above is then applied to solve the time-dependent PDE, in which the spatial derivative operators in the PDE are discretized by these differentiation matrices. Thus, the collocation method requires the PDE to be satisfied only at the collocation points. For non-periodic problems, the collocation points are often chosen to be the roots or the extrema of the Chebyshev or Legendre polynomials, and the interpolants $\phi_1, \ldots, \phi_n$ are chosen to be polynomials. For periodic problems, the collocation points are often chosen to be equispaced points, and the interpolants are chosen to be translates of the sinc function.

The collocation method amounts to the numerical approximation of spatial derivatives of a function f at the collocation points $x_1, \ldots, x_m$ via a global interpolant that is exact at $x_1, \ldots, x_m$, and it can be viewed as high-accuracy limits of finite difference methods (see, for instance, [23]). Compared to the tau and the Galerkin methods, the collocation method often leads to simpler systems of equations to solve, while retaining spectral accuracy (see [11, 17, 30]); and it is relatively easy to apply to PDEs that involve variable coefficients or non-linearities. Therefore, it has become a method of choice for the numerical solution of many types of PDEs, and by now there is an abundance of work in the literature on both its theoretical (see, for example, [29, 64, 65, 67, 68]) and implementational aspects (see, for example, [3, 10, 17, 18, 28, 46]).

Despite its remarkable accuracy and relative simplicity, the collocation method has a principal drawback that limits its applicability in the numerical solution of PDEs. The differentiation matrix D, which takes the values of a function f at the collocation points to the approximate derivatives of f at the same set of points, is ill-conditioned when non-periodic boundary conditions are incorporated. For the case of first derivatives, the differentiation matrix D typically has a spectral radius of size $O(N^2)$, where N is the dimension of D. For the case of second derivatives, the spectral radius becomes $O(N^4)$ (see [64, 68]). This imposes strict stability requirement when the differentiation matrix D is combined with a numerical ODE solver to solve a time-dependent PDE. In many practical situations, the stability of such a combined scheme is determined by the eigenvalues of the differentiation matrix D and the time-step Δt chosen for the ODE solver. More precisely, the eigenvalues of D, multiplied by Δt, have to lie inside the stability region of the ODE solver in order for the scheme to be stable. Therefore, when combining the collocation method with an explicit time-marching scheme, the time-step Δt typically has to be of size $O(N^{-2})$ when solving the hyperbolic PDE $u_t = u_x$, and of size $O(N^{-4})$ when solving the parabolic

PDE $u_t = u_{xx}$. This becomes prohibitive when solving large-scale PDEs, for example, problems in which underlying solutions span large numbers of wavelengths in the spatial domains. On the other hand, using an implicit time-marching scheme usually alleviates the time-step restriction, but since the differentiation matrices arising from the collocation method are dense, a linear or non-linear algebraic system involving a dense matrix has to be solved at each step, which is expensive. In [37], the authors suggested a technique to reduce the spectral radii of the first derivative matrix in the Chebyshev collocation method from $O(N^2)$ to $O(N)$ by first transforming the collocation points. The technique has been applied in a number of situations (see, for example, [4, 18, 32, 50]); however, it requires careful choice of a transformation parameter in order to maintain desired accuracy. Moreover, as pointed out in [50], for practical values of N, the anticipated $O(N^{-1})$ time-step size is not attained, and is replaced by at most doubling of the time-step allowed by the Chebyshev collocation method. Also, the transformation destroys the quadrature properties of the roots of the Chebyshev polynomials, which is undesirable in certain applications (see [13]).

Recently, there has been a growing interest in the development of spectral and pseudospectral methods based on the *prolate spheroidal wave functions* (PSWFs), which were introduced in a series of work by Slepian et. al in the context of the analysis of bandlimited functions defined on intervals (see [42, 43, 57, 58, 59]). In [69], the authors constructed quadrature and interpolation formulas for bandlimited functions based on the PSWFs, demonstrating that the PSWFs are a natural tool for the design of numerical algorithms for bandlimited functions. In particular, the PSWFs and their variants have been used as basis functions in the construction of spectral and pseudospectral methods for the solution of ODEs and PDEs describing wave phenomena (see, for example, [6, 9, 13, 38, 39, 45, 66]). For instance, the authors in [38, 45] constructed collocation methods with the PSWFs as basis functions to solve the Schrödinger's equation, and the authors in [13] constructed collocation methods based on quadrature nodes and roots associated with the PSWFs, and apply them to the solution of hyperbolic PDEs. More recently, the authors in [6] developed a two-dimensional solver for wave equations that involves spatial derivative operators constructed using bases of "approximate" PSWFs. All of the above results reinforced the observations in [69] that for problems involving bandlimited functions, collocation methods based on the PSWFs require fewer points per wavelength to achieve the same accuracy when compared to collocation methods based on orthogonal polynomials, such as Chebyshev or Legendre polynomials. However, while these results are encouraging, the development of pseudospectral methods based on the PSWFs is still in its nascent stage. First, further investigations into the relationship between the choice of the PSWFs in the approximation (1.1) and the resulting accuracies in numerical differentiation and solution of PDEs are needed. Second, while preliminary results in [9, 13, 39, 69] indicate that collocation methods based on PSWFs lead to differentiation matrices of somewhat smaller condition numbers compared to those based on Chebyshev or Legendre polynomials, a systematic picture on the condition numbers of the differentiation matrices in relation to the choice of the PSWFs and the accuracy requirement is still lacking.

In this paper, we introduce a new class of numerical differentiation schemes constructed using the PSWFs. The schemes are constructed based on the approximation of a function f by a linear combination

of PSWFs:

$$f(x) \approx \sum_{j=0}^{n-1} \alpha_j \psi_j^c(x), \tag{1.2}$$

and differ from existing schemes based on PSWFs in two main aspects. First, the PSWFs in the approximation (1.2) are determined by the choice of the bandlimit parameter c and an accuracy requirement ε. In particular, the number of PSWFs n in (1.2) is determined by c and ε. This is different from schemes in [9, 13, 38, 45], in which the bandlimit parameter c and the number of functions n are independently (and, at times, somewhat arbitrarily) chosen. The second difference is that, as opposed to existing pseudospectral methods, which often force the interpolant of the function f to exactly equal to f at a set of collocation points, we do not force the approximation (1.2) to exactly hold at a set of points in the computation of the coefficients $\alpha_1, \ldots, \alpha_n$. Instead, a least-squares type procedure, based on the quadratures for bandlimited functions constructed in [69], is used to compute the coefficients $\alpha_1, \ldots, \alpha_n$. The end result is an $m \times m$ differentiation matrix that takes the values of the function f on a chosen set of quadrature nodes $x_1, \ldots, x_m$ to the approximate values of the derivatives of f on $x_1, \ldots, x_m$. The only requirement on the quadrature is that it integrates the products of the PSWFs $\psi_0^c, \ldots, \psi_{n-1}^c$ to sufficient accuracy. In particular, we do not require the number of PSWFs n in (1.2) to equal the number of nodes m used in the construction of the differentiation matrix, or prescribe any other functional relationship between n and m.

One benefit of the new class of numerical differentiation schemes, which was indicated in the discussion in [69], is that when dealing with problems involving bandlimited functions, it requires fewer points per wavelength to achieve a prescribed accuracy, compared to schemes based on orthogonal polynomials, such as the Chebyshev collocation method. More importantly, in the solution of time-dependent PDEs with non-periodic boundary conditions, the resulting first and second derivative matrices have spectral radii that grow as c and c^2 respectively, for fixed ε. This corresponds to differentiation matrices with spectral radii that grow as m and m^2 respectively for large m, with m being the dimensions of the matrices. The result means that, when we combine these differentiation matrices with an explicit time-marching scheme to solve the PDE, a larger time-step Δt can be chosen compared to when the collocation method is used, while maintaining stability. Combining with the result that differentiation matrices of smaller dimensions are needed to achieve the same accuracy compared to the collocation method, the new class of differentiation schemes are more efficient in the solution of time-dependent PDEs involving bandlimited functions. The advantage is critical in the case of large-scale PDEs, in which a large number of points are needed to discretize the solution in the spatial domain.

The paper is structured as follows. Section 2 provides the necessary mathematical and numerical preliminaries. Section 3 describes the construction of the new class of numerical differentiation schemes based on the PSWFs, as well as modifications to the schemes when boundary conditions are incorporated. In Section 4, we present numerical results pertaining to the accuracy and stability properties of the schemes, and discuss the results of several numerical experiments when the schemes are applied to the solution of time-dependent PDEs and the associated eigenvalue problems. Finally, Section 5 summarizes the work and discusses possible extensions and generalizations.

2 Mathematical and numerical preliminaries

2.1 Quadrature and interpolation

2.1.1 Generalized Gaussian quadratures

The quadrature rules considered in this paper are of the form

$$\sum_{j=1}^{n} w_j \phi(x_j), \tag{2.1}$$

where the points $x_j \in \mathbb{R}$ and coefficients $w_j \in \mathbb{R}$ are referred to as the nodes and weights of the quadrature, respectively. They serve as approximation to integrals of the form

$$\int_a^b \phi(x)\omega(x)dx, \tag{2.2}$$

where $w : [a, b] \to \mathbb{R}$ is an integrable non-negative function.

Quadratures are typically chosen so that the quadrature (2.1) is equal to the integral (2.2) for some set of functions, commonly polynomials of some fixed order. One main example is the classical Gaussian quadrature, which consists of n nodes and integrates polynomials of degree up to $2n - 1$ exactly. The notion of Gaussian quadratures can be generalized to other systems of functions as follows (see [14, 47, 70]):

Definition 2.1. A Gaussian quadrature for a set of $2n$ functions $\phi_1, \ldots, \phi_{2n} : [a, b] \to \mathbb{R}$ with respect to a weight function $\omega : [a, b] \to \mathbb{R}^+$ is a quadrature rule with n weights and nodes that integrates exactly ϕ_i with respect to ω for all $i = 1, \ldots, 2n$. The weights and nodes of a Gaussian quadrature will be referred to as Gaussian weights and nodes, respectively.

Remark 2.1. While the existence of generalized Gaussian quadratures has been proven for a fairly broad class of systems of functions for more than 100 years (see, for instance, [36, 40, 48, 49]), the constructions found in [25, 35, 36, 40, 48, 49] do not easily yield numerical algorithms for the design of such quadratures. Such algorithms have been constructed recently (see [14, 47, 70]).

2.1.2 Discretization of square integrable functions

We shall say that a quadrature rule with nodes $x_1, \ldots, x_n \in [a, b]$ and positive weights $w_1, \ldots, w_n$ discretizes a collection of square integrable functions $f_1, \ldots, f_m$ defined on the interval $[a, b]$ if it integrates exactly all pairwise products of $f_1, \ldots, f_m$; in other words, if

$$\int_a^b f_i(x) f_j(x) dx = \sum_{l=1}^{n} f_i(x_l) f_j(x_l) w_l \tag{2.3}$$

holds for all $i, j = 1, \ldots, m$.

If $x_1, \ldots, x_n, w_1, \ldots, w_n$ is a quadrature discretizing a collection of functions $f_1, \ldots, f_m$ in $L^2([a,b])$, then the map T from the span S of $f_1, \ldots, f_m$ to the Euclidean space $\mathbb{R}^n$ taking the function f to the vector

$$\begin{pmatrix} f(x_1)\sqrt{w_1} \\ f(x_2)\sqrt{w_2} \\ \vdots \\ f(x_n)\sqrt{w_n} \end{pmatrix} \tag{2.4}$$

is a Hilbert space isomorphism (a bijection which preserves inner products) of the subspace S onto the subspace of the Euclidean space $\mathbb{R}^n$ spanned by the vectors

$$\begin{pmatrix} f_1(x_1)\sqrt{w_1} \\ f_1(x_2)\sqrt{w_2} \\ \vdots \\ f_1(x_n)\sqrt{w_n} \end{pmatrix}, \ldots, \begin{pmatrix} f_m(x_1)\sqrt{w_1} \\ f_m(x_2)\sqrt{w_2} \\ \vdots \\ f_m(x_n)\sqrt{w_n} \end{pmatrix}. \tag{2.5}$$

2.1.3 Stable interpolation on quadrature nodes

If $u_1, \ldots, u_k$ is a collection of orthonormal functions in $L^2([a,b])$, and $x_1, \ldots, x_n, w_1, \ldots w_n$ is a quadrature discretizing $u_1, \ldots, u_k$, then $x_1, \ldots, x_n$ serve as stable interpolation nodes for the span of $u_1, \ldots, u_k$. In particular, for a function f defined on $[a,b]$, we can compute stably the coefficients $\alpha_1, \ldots, \alpha_k$ in the linear combination

$$f = \sum_{i=1}^{k} \alpha_i u_i \tag{2.6}$$

using the values $f_1, \ldots, f_n$ of f at the nodes $x_1, \ldots, x_n$. Let U be the $n \times k$ matrix with entries

$$U_{i,j} = u_j(x_i), \tag{2.7}$$

and let

$$F = (f_1, \ldots, f_n)^T, \tag{2.8}$$

$$\alpha = (\alpha_1, \ldots, \alpha_k)^T, \tag{2.9}$$

then (2.6) implies that

$$F = U\alpha. \tag{2.10}$$

Multiplying both sides of (2.10) by the $n \times n$ diagonal matrix W with entries

$$W_{i,i} = \sqrt{w_i},\ i = 1, \ldots, n, \tag{2.11}$$

we obtain

$$WF = A\alpha, \tag{2.12}$$

where

$$A = WU. \tag{2.13}$$

Since the quadrature $x_1, \ldots, x_n, w_1, \ldots, w_n$ discretizes $u_1, \ldots, u_k$, the matrix A has orthonormal columns. Therefore, (2.12) provides a numerically stable formula to compute the coefficients α:

$$\alpha = A^* W F. \tag{2.14}$$

In other words, α can be obtained by applying to F a diagonal matrix followed by a matrix with orthonormal rows. Values of the function f at any point $x \in [a, b]$ can then be computed using (2.6), provided that a scheme for evaluating the functions $u_1, \ldots, u_k$ at arbitrary points is available.

Remark 2.2. For simplicity of discussion, we have assumed that the representation (2.6) of f holds exactly. In actual computations, we often represent f by a truncated series of the form (2.6) that is accurate to a precision ε. As long as the quadrature rule $x_1, \ldots, x_n, w_1, \ldots, w_n$ discretizes $u_1, \ldots, u_k$, (2.14) is a stable interpolation formula for f that is accurate to ε.

Remark 2.3. At first glance, the above procedure for the computation of α seems rather limited in scope, since it relies on the quadrature being exact for all pairwise products of $u_1, \ldots, u_k$. However, as long as the quadrature is reasonably accurate for all pairwise products of $u_1, \ldots, u_k$, the matrix A in (2.12) is close to having orthonormal columns, and therefore sufficiently well-conditioned. In this case, α can be computed stably by solving (2.12) using the least-squares method.

2.2 Prolate spheroidal wave functions

2.2.1 Basic facts

In this subsection, we summarize some basic facts about PSWFs. Unless stated otherwise, all of these facts can be found in [42, 59, 69].

Given a real $c > 0$, we denote by F_c the operator $L^2([-1, 1]) \to L^2([-1, 1])$ defined by the formula

$$F_c(\phi)(x) = \int_{-1}^{1} e^{icxt} \phi(t) dt. \tag{2.15}$$

Obviously, F_c is compact. We denote the eigenvalues of F_c by $\lambda_0, \lambda_1, \ldots, \lambda_j, \ldots$ such that $|\lambda_{j-1}| \geq |\lambda_j|$ for all integer $j \geq 1$. For each integer $j \geq 0$, we denote by ψ_j the eigenfunction corresponding to λ_j; in other words, the integral equation

$$\lambda_j \psi_j(x) = \int_{-1}^{1} e^{icxt} \psi_j(t) dt \tag{2.16}$$

holds for all $x \in [-1, 1]$. Following [69], we adopt the convention that ψ_j are normalized such that $||\psi_j||_{L^2([-1,1])} = 1$ for all j.

The following theorem summarizes the basic properties of ψ_j and λ_j, and can be found in a slightly different form in [69].

Theorem 2.1. *For any real $c > 0$, the eigenfunctions $\psi_0, \psi_1, \ldots,$ of the operator F_c are purely real, orthonormal, and complete in $L^2([-1,1])$. ψ_j is even for all even j and is odd for all odd j. Each function ψ_j has exactly j simple roots in $(-1,1)$. All eigenvalues λ_j of F_c are non-zero and simple; λ_j is purely real for all even j and is purely imaginary for all odd j; in particular, $\lambda_j = i^j|\lambda_j|$.*

We define the self-adjoint operator $Q_c : L^2([-1,1]) \to L^2([-1,1])$ by the formula

$$Q_c(\phi) = \frac{1}{\pi}\int_{-1}^{1} \frac{\sin(c(x-t))}{x-t}\phi(t)dt. \tag{2.17}$$

A simple calculation shows that

$$Q_c = \frac{c}{2\pi} \cdot F_c^* \cdot F_c, \tag{2.18}$$

and that Q_c has the same eigenfunctions as F_c. Moreover, the jth (in descending order) eigenvalue μ_j of Q_c is related to λ_j by the formula

$$\mu_j = \frac{c}{2\pi} \cdot |\lambda_j|^2. \tag{2.19}$$

The operator Q_c is closely related to the operator $P_c : L^2(\mathbb{R}) \to L^2(\mathbb{R})$ defined by the formula

$$P_c(\phi) = \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\sin(c(x-t))}{x-t}\phi(t)dt, \tag{2.20}$$

which is the well-known orthogonal projection operator onto the space of functions of band limit c defined on $\mathbb{R}$.

The following theorem describes the behavior of the spectrum of Q_c, and can be found in [69]. It is proven in a slightly different form in [44].

Theorem 2.2. *For any $c > 0$ and $0 < \alpha < 1$ the number N of the eigenvalues μ_j that are greater than α satisfies the equation*

$$N = \frac{2c}{\pi} + \left(\frac{1}{\pi^2}\log\frac{1-\alpha}{\alpha}\right)\log(c) + O(\log(c)). \tag{2.21}$$

Equation (2.21) implies that for large $c > 0$, Q_c has about $2c/\pi$ eigenvalues with magnitudes very close to 1, followed by order $\log(c)$ eigenvalues decaying exponentially from 1 to nearly 0; the rest of the eigenvalues are all very close to 0.

The eigenfunctions $\psi_0, \psi_1, \ldots, \psi_j, \ldots$ of Q_c turn out to be the PSWFs, a fact well known from classical mathematical physics (see, for example, [52]). The following theorem formalizes this statement, and is proven in a more general form in [57].

Theorem 2.3. *For any $c > 0$, there exists a strictly increasing unbounded sequence of positive numbers $\chi_0, \chi_1, \ldots$ such that, for each integer $j \geq 0$, the differential equation*

$$(1-x^2)\psi''(x) - 2x\psi'(x) + (\chi_j - c^2x^2)\psi(x) = 0 \tag{2.22}$$

has a solution that is continuous on $[-1,1]$. Moreover, all such solutions are constant multiples of ψ_j (2.16).

Remark 2.4. To be more precise, the eigenfunctions $\psi_0, \psi_1, \ldots$ corresponding to the operator F_c (and Q_c) should be denoted $\psi_0^c, \psi_1^c, \ldots$. For simplicity of notation, we will omit the superscript c wherever the value of c is clear from the context.

2.2.2 Approximations of bandlimited functions by PSWFs

In this subsection, we define bandlimited functions and provide a brief review on their approximation by PSWFs. More details can be found in [56, 69].

Definition 2.2. Let $\mathbf{I}$ be an interval in the real line. A function $f : \mathbf{I} \to \mathbb{R}$ is said to be bandlimited if there exists a positive real c and a function $\sigma \in L^2([-1, 1])$ such that

$$f(x) = \int_{-1}^{1} e^{icxt} \sigma(t) dt \tag{2.23}$$

for all $x \in \mathbf{I}$. Moreover, in this case f is said to have bandlimit c.

Obviously, (2.16) implies that ψ_j^c is bandlimited by c for all integer $j \geq 0$.

Since the PSWFs $\psi_0^c, \psi_1^c, \ldots$ constitute a complete orthonormal basis in $L^2([-1, 1])$ (see Theorem 2.1), a function $f : [-1, 1] \to \mathbb{R}$ with bandlimit c has the infinite series expansion

$$f(x) = \sum_{j=0}^{\infty} \alpha_j \psi_j^c(x), \tag{2.24}$$

where the coefficients α_j are given by the formula

$$\alpha_j = \int_{-1}^{1} \psi_j^c(x) f(x) dx, \tag{2.25}$$

for all integer $j \geq 0$. The series (2.24) is convergent in L^2; moreover, it is shown in [56] that the series in (2.24) converges uniformly to f on $[-1, 1]$. The following lemma, which is proven in a slightly different form in [56], provides a bound for approximating a bandlimited function with a truncated expansion in PSWFs.

Lemma 2.4. *Suppose that $f : [-1, 1] \to \mathbb{R}$ has bandlimit c and is expressible as (2.23) and (2.24). Then, for any non-negative integer k, the bound*

$$\left| f(x) - \sum_{j=0}^{k} \alpha_j \psi_k^c(x) \right| \leq \left(\sum_{j=k+1}^{\infty} |\lambda_j^c| (\xi_j^c)^2 \right) \int_{-1}^{1} |\sigma(t)| dt \tag{2.26}$$

holds for all $x \in [-1, 1]$, where

$$\xi_j^c = \max_{-1 \leq x \leq 1} |\psi_j^c(x)|, \ j = 1, 2, \ldots. \tag{2.27}$$

The bound (2.26) indicates that the error is roughly of the order of $|\lambda_k|$, provided that k is in the range where the eigenvalues of Q_c are decaying exponentially (see Theorem 2.2). The numerical results in [69] confirm this observation.

2.2.3 Quadratures for bandlimited functions

We shall say that a quadrature rule with nodes $x_1, \ldots, x_n \in [-1, 1]$ and positive weights $w_1, \ldots, w_n$ integrate functions on $[-1, 1]$ with bandlimit c to precision ε if for any function f on $[-1, 1]$ of the form (2.23), we have

$$\left| \int_{-1}^{1} f(x)dx - \sum_{j=1}^{n} f(x_j)w_j \right| < \varepsilon \int_{-1}^{1} |\sigma(t)|dt. \tag{2.28}$$

One procedure to construct Gaussian quadratures for bandlimited functions is to use the Newton-type nonlinear optimization algorithm of [14]. Specifically, for bandlimit c and precision ε, the algorithm constructs an $n/2$-point Gaussian quadrature integrating exactly the first n PSWFs $\psi_0^c, \psi_1^c, \ldots, \psi_{n-1}^c$, where n is the smallest integer such that the corresponding eigenvalue has magnitude less than ε, i.e.,

$$|\lambda_{n-1}| \geq \varepsilon > |\lambda_n|. \tag{2.29}$$

The constructed quadrature then integrates all functions on $[-1, 1]$ with bandlimit c to precision ε, provided n is in the range where the eigenvalues of Q_c decay exponentially.

Remark 2.5. The procedure described above is expensive, and requires order n^3 operations (see [14]). An alternative procedure is described in [69], and is based on a generalization of the Euclid's division algorithm to PSWFs. While it is less expensive than the procedure of [14], the quadrature constructed is a bit less accurate. Numerical examples demonstrating the performance of quadratures constructed by both procedures can be found in [69].

2.2.4 Interpolation of bandlimited functions

As discussed in Section 2.2.2, given a function $f : [-1, 1] \to \mathbb{R}$ with bandlimit c, we can approximate it by a linear combination of the first n PSWFs $\psi_0^c, \psi_1^c, \ldots, \psi_{n-1}^c$:

$$\sum_{j=0}^{n-1} \alpha_j \psi_j^c, \tag{2.30}$$

with an error roughly of the order of $|\lambda_n|$, provided that n is in the range where the eigenvalues of Q_c decay exponentially. Given a quadrature $x_1, \ldots, x_m, w_1, \ldots, w_m$ that integrates exactly all pairwise products of $\psi_0^c, \ldots, \psi_{n-1}^c$, we can apply the same argument as in Section 2.1.3 to express the problem of computing the coefficients α_j as

$$WF = A\alpha, \tag{2.31}$$

where

$$F = (f(x_1), \ldots, f(x_m))^T, \tag{2.32}$$

$$\alpha = (\alpha_0, \ldots, \alpha_{n-1})^T, \tag{2.33}$$

W is an $m \times m$ diagonal matrix with entries

$$W_{i,i} = \sqrt{w_i},\ i = 1, \ldots, m, \tag{2.34}$$

and A is an $m \times n$ matrix with entries

$$A_{i,j} = \sqrt{w_i}\psi_j^c(x_i). \tag{2.35}$$

The matrix A has orthonormal columns, and α can be computed stably using the formula

$$\alpha = A^* W F. \tag{2.36}$$

(2.30) then gives an interpolation formula for f with error proportional to $|\lambda_n|$.

Remark 2.6. Theorem 7.1 of [69] shows that a quadrature integrating all functions on $[-1, 1]$ of bandlimit $2c$ to precision $|\lambda_n^c|^2$ guarantees that the matrix A in equation (2.31) is close to having orthonormal columns, and hence sufficiently well-conditioned for the accurate computation of α (see Remark 2.3). In the numerical experiments in Section 4.1.1, we use a quadrature corresponding to bandlimit $2c$ and precision $|\lambda_n^c|^2 \times 10^{-10}$ in order to ensure that A has orthonormal columns.

Remark 2.7. Both the procedures of [14] and [69] (see Remark 2.5) can be used to construct quadratures for the interpolation of bandlimited functions. In addition, the numerical results in [69] show that the two quadratures have virtually identical performance when used for interpolation (as opposed to integration), achieving the same accuracy with the same number of nodes.

2.3 Numerical differentiation

2.3.1 Finite difference method

The finite difference method is a classical numerical differentiation method based on local polynomial interpolation of a function at equispaced points. Given a function $f : \mathbb{R} \to \mathbb{R}$ and $h \neq 0$, the Taylor expansion of $f(x + h)$ around the point x:

$$f(x + h) = f(x) + \frac{f'(x)}{h} + O(h^2) \tag{2.37}$$

gives rise to the following one-sided difference formula for the approximation of $f'(x)$:

$$T_1(h) = \frac{f(x + h) - f(x)}{h}, \tag{2.38}$$

which has an error of $O(h)$. Note that formula (2.38) also results from linear interpolation of f at the points x and $x+h$. By constructing the degree-two polynomial p that interpolates f at the points $x-h$, x, and $x+h$, and then computing $p'(x)$, we obtain the centered difference formula for $f'(x)$:

$$T_2(h) = \frac{f(x+h) - f(x-h)}{2h}, \tag{2.39}$$

which has an error of $O(h^2)$; in other words, (2.39) is second-order accurate. Alternatively, (2.39) is derived by combining the Taylor expansions of $f(x+h)$ and $f(x-h)$ around the point x.

Applying the same idea of local polynomial interpolation, we obtain finite difference formulas with higher orders of accuracy, or for higher-order derivatives. Table 1 lists, with corresponding orders of accuracy, some finite difference formulas for the approximations of f' and f'' using the values of f at equispaced points of step-size h. More comprehensive tables of finite difference formulas can be found, for instance, in [22, 23].

Table 1: Finite difference formulas for the approximations of f' and f'' using the values of f at equispaced points.

Derivative	Finite difference formula	Order of accuracy
$f'(x)$	$\frac{f(x+h)-f(x-h)}{2h}$	2
$f'(x)$	$\frac{-3f(x)+4f(x+h)-f(x+2h)}{2h}$	2
$f'(x)$	$\frac{f(x-2h)-8f(x-h)+8f(x+h)-f(x+2h)}{12h}$	4
$f'(x)$	$\frac{-3f(x-h)-10f(x)+18f(x+h)-6f(x+2h)+f(x+3h)}{12h}$	4
$f'(x)$	$\frac{-25f(x)+48f(x+h)-36f(x+2h)+16f(x+3h)-3f(x+4h)}{12h}$	4
$f''(x)$	$\frac{f(x-h)-2f(x)+f(x+h)}{h^2}$	2
$f''(x)$	$\frac{2f(x)-5f(x+h)+4f(x+2h)-f(x+3h)}{h^2}$	2
$f''(x)$	$\frac{-f(x-2h)+16f(x-h)-30f(x)+16f(x+h)-f(x+2h)}{12h^2}$	4
$f''(x)$	$\frac{10f(x-h)-15f(x)-4f(x+h)+14f(x+2h)-6f(x+3h)+f(x+4h)}{12h^2}$	4
$f''(x)$	$\frac{45f(x)-154f(x+h)+214f(x+2h)-156f(x+3h)+61f(x+4h)-10f(x+5h)}{12h^2}$	4

2.3.2 Chebyshev collocation method

In this subsection, we briefly outline the Chebyshev collocation method for numerical differentiation. Detailed discussions of its implementation, as well as its accuracy and stability properties, can be found in, for instance, [8, 23, 30, 63].

Consider the Chebyshev-Gauss-Lobatto collocation points $x_1, \ldots, x_m$ defined on $[-1, 1]$ via the formula

$$x_i = \cos\left(\frac{\pi(i-1)}{m-1}\right), \ i = 1, \ldots, m. \tag{2.40}$$

In particular, $x_1, \ldots, x_m$ are arranged in descending order, and we have $x_1 = 1$ and $x_m = -1$. The

points $x_1, \ldots, x_m$ are the extrema of the Chebyshev polynomial of order $m-1$:

$$T_{m-1}(x) = \cos\left((m-1)\cos^{-1} x\right). \tag{2.41}$$

Given a function $f : [-1, 1] \to \mathbb{R}$, the Chebyshev collocation method approximates the derivatives of f at the collocation points $x_1, \ldots, x_m$ by first constructing the interpolating polynomial f_m of degree $m-1$ that agrees with f at $x_1, \ldots, x_m$, i.e.,

$$f_m(x) = \sum_{j=1}^{m} f(x_j) g_j(x), \tag{2.42}$$

where g_j is a polynomial of degree $m-1$ uniquely defined by its values at $x_1, \ldots, x_m$:

$$g_j(x_k) = \begin{cases} 1 & \text{if } j = k, \\ 0 & \text{if } j \neq k. \end{cases} \tag{2.43}$$

It can be shown (see, for example, [8]) that

$$g_j(x) = \frac{(-1)^j (1-x^2) T'_{m-1}(x)}{c_j (m-1)^2 (x - x_j)}, \ j = 1, \ldots, m, \tag{2.44}$$

where

$$c_j = \begin{cases} 2 & \text{if } j = 1, m, \\ 1 & \text{if } j = 2, \ldots, m-1. \end{cases} \tag{2.45}$$

The interpolating polynomial (2.42) provides approximations $d_1, \ldots, d_m$ of the first derivatives of f at $x_1, \ldots, x_m$ by the formula

$$d_i = \sum_{j=1}^{m} f(x_j) g'_j(x_i), \ i = 1, \ldots, m. \tag{2.46}$$

Therefore, the $m \times m$ Chebyshev differentiation matrix D with entries $D_{i,j} = g'_j(x_i)$ is a linear operator taking the values $f_1, \ldots, f_m$ of the function f at the collocation points $x_1, \ldots, x_m$ to the approximate values $d_1, \ldots, d_m$ of f' at $x_1, \ldots, x_m$. A straightforward calculation gives us an explicit formula for the entries of D:

$$D_{i,j} = \frac{c_i}{c_j} \frac{(-1)^{i+j}}{x_i - x_j}, \ i \neq j, \tag{2.47}$$

$$D_{i,i} = -\frac{1}{2} \frac{x_i}{1 - x_i^2}, \ i \neq 1, m, \tag{2.48}$$

$$D_{1,1} = \frac{2(m-1)^2 + 1}{6} = -D_{m,m}. \tag{2.49}$$

Alternatively, since the interpolating polynomial (2.42) to the vector $(1, \ldots, 1)^T$ is exactly the constant function $f(x) = 1$, and that $f'(x) = 0$ for all x, the matrix D must map $(1, \ldots, 1)^T$ to the zero vector,

by construction. Thus, we can compute the diagonal entries of D by the formula

$$D_{i,i} = \sum_{j=1, j\neq i}^{m} D_{i,j},\ i = 1, \ldots, m. \tag{2.50}$$

The reduction of rounding errors in computing the diagonal entries of D via (2.50) instead of (2.48)-(2.49) are discussed in [2, 3].

By the same argument, the $m \times m$ Chebyshev differentiation matrix $\tilde{D}$ for the approximation of second derivatives of functions $f : [-1, 1] \to \mathbb{R}$ has entries $g_j''(x_i)$. The off-diagonal entries of $\tilde{D}$ are given by the formula

$$\tilde{D}_{i,j} = \frac{1}{c_j}(-1)^{i+j+1}\frac{2 - x_i x_j - x_i^2}{(1 - x_i^2)(x_i - x_j)^2},\ i \neq j;\ i \neq 1, m, \tag{2.51}$$

$$\tilde{D}_{1,j} = \frac{(-1)^j}{c_j}\left(\frac{-4(m-1)^2 - 2}{3(x_1 - x_j)} + \frac{4}{(x_1 - x_j)^2}\right),\ j \neq 1, \tag{2.52}$$

$$\tilde{D}_{m,j} = \frac{(-1)^j}{c_j}\left(\frac{(-1)^{m-1}(4(m-1)^2 + 2)}{3(x_m - x_j)} + \frac{(-1)^{m-1}4}{(x_m - x_j)^2}\right),\ j \neq m. \tag{2.53}$$

The diagonal entries of $\tilde{D}$ are then computed by the formula

$$\tilde{D}_{i,i} = \sum_{j=1, j\neq i}^{m} \tilde{D}_{i,j},\ i = 1, \ldots, m. \tag{2.54}$$

Remark 2.8. Since the distance between the Chebyshev collocation points $x_1, \ldots, x_m$ near the ends of the interval $[-1, 1]$ grow as m^2 (see, for example, [23]), (2.47) involves division by small numbers when m is large. As discussed in [17], this is remedied by using the trigonometric identities

$$x_i - x_j = 2\sin\left((i + j - 2)\frac{\pi}{2(m-1)}\right)\sin\left((j - i)\frac{\pi}{2(m-1)}\right),\ i \neq j, \tag{2.55}$$

to rewrite (2.47) into the form

$$D_{i,j} = -\frac{c_i}{2c_j}\frac{(-1)^{i+j}}{\sin\left((i + j - 2)\frac{\pi}{2(m-1)}\right)\sin\left((i - j)\frac{\pi}{2(m-1)}\right)},\ i \neq j. \tag{2.56}$$

In the numerical experiments of this paper, (2.56) is used instead of (2.47) to compute the off-diagonal entries of D. The off-diagonal entries of $\tilde{D}$ are computed using a similar reformulation of (2.51)-(2.53).

3 Numerical differentiation via PSWFs

3.1 Differentiation of bandlimited functions

For any integer $k \geq 1$, the kth order derivative of a bandlimited function $f : [-1, 1] \to \mathbb{R}$ can be computed numerically by approximating f by a linear combination of PSWFs, and then differentiating the PSWFs k times. In this subsection, we construct the numerical differentiation scheme for computing the second derivative of f using the PSWFs. The other derivatives of f can be computed in the exact same manner.

Let $f : [-1, 1] \to \mathbb{R}$ be a bandlimited function with bandlimit c, and fix some small $\varepsilon > 0$. We first consider the approximation of f by the linear combination of the first n PSWFs:

$$\sum_{j=0}^{n-1} \alpha_j \psi_j^c, \tag{3.1}$$

where n is chosen to be the smallest integer such that the corresponding eigenvalue of the operator F_c has magnitude smaller than ε, i.e.,

$$|\lambda_{n-1}| \geq \varepsilon > |\lambda_n|. \tag{3.2}$$

Next, we compute the coefficients α_j in (3.1). First, we apply the algorithm of [69] to construct a quadrature $x_1, \ldots, x_m, w_1, \ldots, w_m$ that integrates exactly all pairwise products of $\psi_0, \ldots, \psi_{n-1}$ (see Remark 2.6). Then, we evaluate the values $f_1, \ldots, f_m$ of f at the nodes $x_1, \ldots, x_m$, and apply formula (2.36) to obtain

$$\alpha = P^* W F, \tag{3.3}$$

where

$$F = (f_1, \ldots, f_m)^T, \tag{3.4}$$

$$\alpha = (\alpha_0, \ldots, \alpha_{n-1})^T, \tag{3.5}$$

W is an $m \times m$ diagonal matrix with entries $W_{i,i} = \sqrt{w_i}$, and P is an $m \times n$ matrix with entries

$$P_{i,j} = \sqrt{w_i} \psi_{j-1}(x_i), \ i = 1, \ldots, m; \ j = 1, \ldots, n. \tag{3.6}$$

Once α is computed, we can approximate the value of f'' on any $x \in [-1, 1]$ by the formula:

$$\sum_{j=0}^{n-1} \alpha_j \psi_j'', \tag{3.7}$$

using a numerical scheme for evaluating ψ_j'' at arbitrary points (see [69]).

In particular, the $m \times m$ matrix D defined by the formula

$$D = U P^* W, \tag{3.8}$$

where U is an $m \times n$ matrix with entries

$$U_{i,j} = \psi''_{j-1}(x_i),\ i = 1, \ldots, m;\ j = 1, \ldots, n, \tag{3.9}$$

is a linear operator taking the values $f_1, \ldots, f_m$ of the function f at the interpolation nodes $x_1, \ldots, x_m$ to the values $d_1, \ldots, d_m$, where d_j is the approximation to $f''(x_j)$ via (3.7). We refer to D as a *differentiation matrix* constructed using the PSWFs. In Section 4.1.1, we demonstrate the accuracy property of the above scheme via several numerical experiments.

Remark 3.1. For the algorithm of [69], the nodes of the quadrature constructed for bandlimit $2c$ and precision ε_q are precisely the roots of ψ_l^c, where l is roughly half of the number of PSWFs ψ_j^{2c} with eigenvalues greater than ε_q. Therefore, the quadrature nodes constructed by the algorithm of [69], and hence the interpolation nodes on which the differentiation matrix D is constructed, all lie in the interior of the interval $[-1, 1]$.

Remark 3.2. Clearly, the scheme described above is not restricted to functions defined on $[-1, 1]$. For example, consider a function $g : [a, b] \to \mathbb{R}$ defined by transforming another function $f : [-1, 1] \to \mathbb{R}$ by the formula

$$g(x) = f((2x - a - b)/(b - a)),\ x \in [a, b]; \tag{3.10}$$

in particular, if f is a periodic function with K wavelengths on $[-1, 1]$, then g is also periodic, with K wavelengths on $[a, b]$. Suppose a differentiation matrix D (3.8) that approximates f'' at the nodes $x_1, \ldots, x_m \in [-1, 1]$ is constructed using the above scheme, then the matrix $\tilde{D}$ obtained by the formula

$$\tilde{D}_{i,j} = (2/(b - a))^2 D_{i,j},\ i, j = 1, \ldots, m, \tag{3.11}$$

approximates g'' at the nodes $y_1, \ldots, y_m \in [a, b]$ defined by

$$y_i = (b - a)x_i/2 + (a + b)/2,\ i = 1, \ldots, m, \tag{3.12}$$

to the same order of relative accuracies. For the case of kth derivative, the same argument applies with the exponent 2 in (3.11) replaced by k.

3.2 Differentiation matrices incorporating boundary conditions

For almost all kinds of PDEs encountered in practice, boundary conditions are needed to guarantee the uniqueness of solutions. The numerical solutions of these PDEs usually require the incorporation of the boundary conditions into the discretized systems. One approach of incorporating boundary conditions for spectral (and pseudospectral) methods is to modify the interpolation functions to make them satisfy the boundary conditions; another approach is to add additional equations to the discretized systems to enforce the boundary conditions, without modifying the interpolation functions (see, for instance, [8, 23, 30, 63]). In this subsection, we adopt the first approach, and modify the scheme in Section 3.1 to construct differentiation matrices that incorporate boundary conditions.

As a specific example, we consider the wave equation with the homogeneous Dirichlet boundary condition:

$$\begin{aligned} u_{tt} &= u_{xx}, & x \in (-1,1), \quad t > 0, \\ u(x,0) &= f(x), & x \in [-1,1], \\ u(-1,t) &= u(1,t) = 0, & t \geq 0, \end{aligned} \tag{3.13}$$

and construct a differentiation matrix D that discretizes the derivative operator u_{xx} and incorporates the boundary condition. The main objective is to construct using the PSWFs an orthonormal set of interpolation functions $\phi_1, \ldots, \phi_k$ in $L^2([-1,1])$ such that each ϕ_j satisfies the zero boundary condition:

$$\phi_j(-1) = \phi_j(1) = 0, \quad j = 1, \ldots, k. \tag{3.14}$$

First, we fix a bandlimit c and precision ε, and take the first n PSWFs $\psi_0, \ldots, \psi_{n-1}$, with n chosen to be the smallest integer such that $|\lambda_n| < \varepsilon$. We then subtract from each ψ_j a linear function μ_j interpolating the values of ψ_j at the endpoints of $[-1,1]$:

$$\mu_j(x) = \psi_j(-1) + (\psi_j(1) - \psi_j(-1))\frac{x+1}{2}. \tag{3.15}$$

The resulting set of functions

$$\tilde{\psi}_j = \psi_j - \mu_j, \quad j = 0, \ldots, n-1, \tag{3.16}$$

approximate, with an error of the order of ε, all functions $f : [-1,1] \to \mathbb{R}$ that have bandlimit c and satisfy the zero boundary condition $f(-1) = f(1) = 0$.

Next, we construct an orthonormal basis for the span of $\tilde{\psi}_0, \ldots, \tilde{\psi}_{n-1}$. We first apply the algorithm of [69] to construct a quadrature $x_1, \ldots, x_m, w_1, \ldots, w_m$ that discretizes $\tilde{\psi}_0, \ldots, \tilde{\psi}_{n-1}$ (see Remark 3.6 below). Then, we apply the pivoted Gram-Schmidt algorithm (with re-orthogonalization) described in [7] to the $m \times n$ matrix A with entries

$$A = \begin{pmatrix} \tilde{\psi}_0(x_1)\sqrt{w_1} & \tilde{\psi}_1(x_1)\sqrt{w_1} & \ldots & \tilde{\psi}_{n-1}(x_1)\sqrt{w_1} \\ \tilde{\psi}_0(x_2)\sqrt{w_2} & \tilde{\psi}_1(x_2)\sqrt{w_2} & \ldots & \tilde{\psi}_{n-1}(x_2)\sqrt{w_2} \\ \ldots & \vdots & \ddots & \vdots \\ \tilde{\psi}_0(x_m)\sqrt{w_m} & \tilde{\psi}_1(x_m)\sqrt{w_m} & \ldots & \tilde{\psi}_{n-1}(x_m)\sqrt{w_m} \end{pmatrix}, \tag{3.17}$$

obtaining an $m \times n$ matrix $\tilde{A}$. Since $\psi_0, \ldots, \psi_{n-1}$ approximates all bandlimited functions of bandlimit c to precision ε, and $\tilde{\psi}_0, \ldots, \tilde{\psi}_{n-1}$ are obtained by subtracting from $\psi_0, \ldots, \psi_{n-1}$ two degrees of freedom, the matrix A has ε-rank $k = n - 2$. We take the first k columns of $\tilde{A}$ and define the interpolation functions $\phi_1, \ldots, \phi_k$ by their values:

$$\phi_j(x_i) = \tilde{A}_{i,j}/\sqrt{w_i}, \ i = 1, \ldots, m, \ j = 1, \ldots, k \tag{3.18}$$

at the quadrature nodes $x_1, \ldots, x_m$.

Remark 3.3. Although $\phi_1, \ldots, \phi_k$ are defined by their values on the quadrature nodes $x_1, \ldots, x_m$, their

values on any arbitrary point $x \in [-1, 1]$ are completely determined by (3.16) and the pivoted Gram-Schmidt procedure. Specifically, let L be the $n \times n$ linear transformation corresponding to the pivoted Gram-Schmidt procedure:

$$\tilde{A} = AL, \tag{3.19}$$

then the value of $\phi_i(x)$ is obtained by applying L from the right to the row vector

$$(\tilde{\psi}_0(x), \ldots, \tilde{\psi}_{n-1}(x)), \tag{3.20}$$

and taking the ith element of the resulting row vector. In particular, each of the ϕ_i satisfies the boundary condition (3.14).

The remaining part of the construction is similar to that of Section 3.1. Let $f : [-1, 1] \to \mathbb{R}$ be a function of bandlimit c such that $f(-1) = f(1) = 0$, then f can be approximated by a linear combination of $\phi_1, \ldots, \phi_k$:

$$\sum_{j=1}^{k} \alpha_j \phi_j, \tag{3.21}$$

with an error of the order of ε. We use the quadrature nodes $x_1, \ldots, x_m$ as interpolation nodes, and compute the $m \times m$ differentiation matrix D taking the values $(f(x_1), \ldots, f(x_m))$ to the approximating values $(d_1, \ldots, d_m)$ of $(f''(x_1), \ldots, f''(x_m))$ by the formula

$$D = UP^*W, \tag{3.22}$$

where W is an $m \times m$ diagonal matrix with diagonal entries $W_{i,i} = \sqrt{w_i}$, P is an $m \times k$ matrix with entries

$$P_{i,j} = \phi_j(x_i)\sqrt{w_i},\ i = 1, \ldots, m;\ j = 1, \ldots, k, \tag{3.23}$$

and U is an $m \times k$ matrix with entries

$$U_{i,j} = \phi_j''(x_i),\ i = 1, \ldots, m;\ j = 1, \ldots, k. \tag{3.24}$$

With suitably chosen c and ε, the differentiation matrix D can be combined with a suitable time-marching scheme to solve (3.13) to any desired accuracy.

Remark 3.4. The entries (3.24) of U are obtained by applying the matrix L in (3.19) from the right to the $m \times n$ matrix S defined by

$$S_{i,j} = \tilde{\psi}_{j-1}''(x_i),\ i = 1, \ldots, m,\ j = 1, \ldots, n, \tag{3.25}$$

and then taking the first k columns of SL. In particular, the matrices A and S are passed together to the pivoted Gram-Schmidt algorithm during actual implementation of the scheme.

Differentiation matrices that incorporate other boundary conditions, such as the Neumann boundary

condition:

$$u_x(-1,t) = 0, \quad u(1,t) = 0, \quad t \geq 0, \tag{3.26}$$

are similarly constructed by modifying the functions μ_j in (3.15). Likewise, differentiation matrices for computing first derivatives are constructed with the straightforward modification of the above scheme. In Section 4.2, we demonstrate the performance of the scheme by applying the differentiation matrices constructed to the solution of several time-dependent PDEs and the associated eigenvalue problems.

Remark 3.5. For any time $T > 0$, we combine the differentiation matrix D with a time-marching scheme to solve (3.13), obtaining numerical approximation $(u_1, \ldots, u_m)$ to $u(\cdot, T)$ at the nodes $x_1, \ldots, x_m$. Numerical approximation to $u(\cdot, T)$ at another set of nodes $y_1, \ldots, y_l$ are then computed easily by an interpolation scheme based on (3.21) and (2.14).

Remark 3.6. In order to construct $\phi_1, \ldots, \phi_k$ by the Gram-Schmidt algorithm, the quadrature $x_1, \ldots, x_m$, $w_1 \ldots, w_m$ has to be chosen such that it discretizes $\tilde{\psi}_0, \ldots, \tilde{\psi}_{n-1}$. In the numerical experiments in Section 4.1.2, we construct using the algorithm of [69] a quadrature integrating all functions on $[-1, 1]$ of bandlimit $2c$ to precision ε^2. Other quadratures, such as the Gaussian-Legendre quadrature, can also be used. As observed in [69], however, the latter choice requires more nodes to achieve the same accuracy.

Remark 3.7. The non-zero eigenvalues of the differentiation matrix D constructed by the above scheme depends only on the PSWFs $\psi_0, \ldots, \psi_{n-1}$, and hence only on the choices of c and ε. In particular, they do *not* depend on the particular quadrature used or its number of nodes m, as long as it is accurate enough to discretize the functions $\tilde{\psi}_0, \ldots, \tilde{\psi}_{n-1}$. In other words, given fixed c and ε and an appropriately chosen quadrature, the non-zero part of the spectrum of D is independent of its dimension m.

4 Numerical results

4.1 Accuracy and stability properties

In this subsection, we demonstrate results regarding the accuracy and stability properties of the numerical differentiation schemes described in Section 3. The schemes were implemented in FORTRAN 77. All implementations were carried out in double (16-digit) precision arithmetic except otherwise noted.

4.1.1 Accuracies of the differentiation matrices

In this subsection, we present results related to the accuracy of the differentiation matrices constructed by the algorithm described in Section 3.1. Let D be the $m \times m$ differentiation matrix constructed using the algorithm with bandlimit c and precision ε, taking the values of functions $f : [-1, 1] \to \mathbb{R}$ at the interpolation nodes $x_1, \ldots, x_m \in [-1, 1]$ to its approximate derivatives at the nodes. For each $a \geq 0$, we apply D to obtain approximations of the derivatives of the functions $\sin(ax)$ and $\cos(ax)$ at $x_1, \ldots x_m$. We denote the approximations by $u_1, u_2, \ldots, u_{2m}$ and the corresponding analytical derivatives

by $v_1, v_2, \ldots, v_{2m}$, and measure the errors using the following three quantities:

$$E_{a,1} = \max_{1 \le i \le 2m} |u_i - v_i|, \tag{4.1}$$

$$E_{a,2} = \sqrt{\frac{\sum_{i=1}^{2m} (u_i - v_i)^2}{2m}}, \tag{4.2}$$

$$E_{a,3} = \sqrt{\frac{\sum_{i=1}^{2m} (u_i - v_i)^2}{\sum_{i=1}^{2m} v_i^2}}. \tag{4.3}$$

In other words, $E_{a,1}$, $E_{a,2}$, and $E_{a,3}$ are the maximum absolute errors, root-mean-squared errors, and relative l_2 errors respectively. Table 2 shows, for matrices D constructed using $\varepsilon = 10^{-14}$ and varying values of c, the maximum errors $\max_{a \in [0,c]} E_{a,1}$ and $\max_{a \in [0,c]} E_{a,2}$ over a large number of values of a sampled on $[0, c]$, as well as the relative l_2 error $E_{c,3}$. Table 3, on the other hand, shows the above errors for a fixed bandlimit $c = 1000$ and varying values of ε. In the tables, m denotes the number of nodes $x_1, \ldots, x_m$ used in the construction of D, and hence the dimension of D, while n denotes the number of interpolating PSWFs ψ_j^c corresponding to the choice of c and ε (see (3.1)-(3.2)). Figure 1 shows the absolute errors across the nodes on $[-1, 1]$ when D is used to differentiate the single function $f(x) = \sin(cx)$, where D is constructed with $\varepsilon = 10^{-14}$ and $c = 50, 1200$. Lastly, Figure 2 shows, for D constructed using $c = 50, 100$ and $\varepsilon = 10^{-8}, 10^{-14}$, the errors $E_{a,1}$ across a large number of values of a sampled on $[0, 200]$. Results associated with both first and second derivatives are shown in the above tables and figures. In the following discussion, we further explain our results and make several observations.

1. In these experiments, the nodes $x_1, \ldots, x_m$ are chosen to be the nodes of a quadrature integrating functions of bandlimit $2c$ to precision $\varepsilon^2 \times 10^{-10}$, constructed using the algorithm of [69]. This ensures that the columns of the matrix P (3.6) are orthonormal, to machine precision. From inspection of Theorem 2.2, the dependence of the number of PSWFs ψ_j^{2c} on the eigenvalue cutoff ε is weak. Hence, choosing the precision $\varepsilon^2 \times 10^{-10}$ leads to only slightly larger numbers of interpolation nodes compared to those used in the numerical experiments of [69] (see Remark 3.1).

2. As expected again from Theorem 2.2, the dependence of the number of nodes m on the precision ε is weak. For example, Table 3 shows that for $c = 1000$, we roughly gain one digit of accuracy by increasing the number of nodes m by about 3.

3. From Table 2, we see that the *sampling factor*, defined as the ratio of m over n, is close to 1, and approaches closer to 1 as c increases. The reason is because m is roughly half of the number of PSWFs ψ_j^{2c} with eigenvalues greater than $\varepsilon^2 \times 10^{-10}$ (see Remark 3.1), and the latter quantity is roughly double that of n, due to Theorem 2.2.

4. For large c, the ratio of the number of nodes m over $\frac{c}{\pi}$ is close to two, regardless of the accuracy requirement. This means that, for large c, a slightly more than two points per wavelength are needed to achieve any desired accuracy.

5. For each $a \geq 0$, the maximum absolute error $E_{a,1}$ always occurs near the ends of the interval $[-1, 1]$, regardless of the differentiation matrix used to approximate the derivatives. Figure 1 shows, as an example, that for first derivatives the accuracies are usually $1 - 2$ digits lower near the endpoints compared to the rest of the interval, while for second derivatives the accuracies are usually $3 - 4$ digits lower near the endpoints. This is attributed to the fact that the magnitudes of the derivatives of each PSWF $\psi_j^c(x)$ rises sharply as x approaches the ends of $[-1, 1]$, which, in turn, implies that the norms of the first and last few rows of the differentiation matrices are usually several orders of magnitude larger than those of the rows near the middle. As an example, Table 4 shows the l_2 norms of the first and middle rows of the differentiation matrices constructed using different values of c and ε.

6. From Table 2, we see that the absolute measures of errors $\max_{a\in[0,c]} E_{a,1}$ and $\max_{a\in[0,c]} E_{a,2}$ increase roughly at a rate of c and c^2 for first and second derivatives, respectively. This is consistent with the fact that the amplification factors for taking the first and second derivatives of the functions $\sin(cx)$ and $\cos(cx)$ are about c and c^2, respectively. For example, we see from Figure 1 that for $c = 1200$ and $\varepsilon = 10^{-14}$, we get an absolute error of about 10^{-8} near the center of $[-1, 1]$ when approximating the second derivatives of $f(x) = \sin(1200x)$, losing about 6 digits relative to the prescribed ε.

7. From Figure 2, we see that the maximum absolute error $E_{a,1}$ for a differentiation matrix constructed using bandlimit c is almost uniform over all $0 \leq a \leq c$. The accuracy decreases exponentially once a increases beyond the prescribed bandlimit c of the matrix, with the rate of decrease almost the same for $c = 50$ and $c = 100$. Our more extensive tests show that for $\varepsilon = 10^{-14}$ and any prescribed c, no accuracy is obtained when a gets about 5 beyond c. This amounts to fewer than two wavelengths over the interval $[-1, 1]$.

8. From Tables 2 and 3 we see that, given any c, the relative l_2 errors $E_{c,3}$ are consistently about 1 and 2 digits less than the prescribed ε, for first and second derivatives respectively. Figure 3 shows the errors $E_{a,3}$ across $0 \leq a \leq c$ for the second derivative matrix constructed with $c = 1000$ and $\varepsilon = 10^{-8}, 10^{-14}$, indicating that $E_{a,3}$ increases as a decreases towards 0. This is expected, given the observation in (7) that the matrices constructed by the algorithm give roughly uniform absolute errors for all $0 \leq a \leq c$. Therefore, it is optimal in terms of both absolute and relative errors to approximate the derivatives of functions $f : [-1, 1] \to \mathbb{R}$ with bandlimit roughly c with differentiation matrices constructed using c.

c	m	n	First derivatives			Second derivatives		
			$\max_{a\in[0,c]} E_{a,1}$	$\max_{a\in[0,c]} E_{a,2}$	$E_{c,3}$	$\max_{a\in[0,c]} E_{a,1}$	$\max_{a\in[0,c]} E_{a,2}$	$E_{c,3}$
5.0	23	20	7.72E-13	1.74E-13	4.72E-14	1.06E-10	2.61E-11	1.36E-12
10.0	30	26	4.20E-12	7.67E-13	5.25E-14	6.89E-10	1.27E-10	3.04E-13
20.0	41	36	4.51E-12	7.26E-13	2.16E-14	2.16E-09	3.39E-10	5.45E-13
30.0	50	44	8.45E-12	1.34E-12	6.33E-14	5.66E-09	8.41E-10	1.32E-12
40.0	58	52	2.28E-11	3.17E-12	1.01E-13	1.14E-08	1.56E-09	1.34E-12
50.0	66	60	1.88E-11	2.37E-12	6.38E-14	1.11E-08	1.62E-09	9.16E-13
80.0	89	82	4.72E-11	5.30E-12	7.65E-14	4.84E-08	5.85E-09	1.29E-12
100.0	103	96	7.23E-11	7.57E-12	3.23E-14	8.94E-08	8.93E-09	4.83E-13
200.0	173	164	1.45E-10	1.35E-11	7.72E-14	4.80E-07	3.82E-08	8.54E-13
300.0	240	230	3.77E-10	2.63E-11	7.15E-14	1.89E-06	1.42E-07	2.44E-13
400.0	306	294	1.50E-09	8.96E-11	1.36E-13	6.95E-06	4.05E-07	1.68E-12
500.0	372	360	2.00E-09	1.20E-10	1.73E-13	1.24E-05	6.59E-07	2.01E-12
800.0	567	544	3.43E-09	1.54E-10	2.47E-13	4.84E-05	2.10E-06	2.21E-12
1000.0	696	682	2.95E-09	1.57E-10	2.22E-13	5.68E-05	2.25E-06	2.81E-12
1200.0	825	812	2.03E-08	7.24E-10	6.92E-13	2.64E-04	9.41E-06	5.93E-12
1400.0	954	940	9.29E-09	3.57E-10	2.86E-13	1.96E-04	6.51E-06	3.59E-12
1600.0	1085	1068	1.72E-08	5.59E-10	4.58E-13	3.75E-04	1.39E-05	2.55E-12
1800.0	1213	1196	1.50E-08	4.62E-10	2.34E-13	5.53E-04	1.86E-05	1.07E-12
2000.0	1342	1324	2.06E-08	5.96E-10	1.90E-13	5.10E-04	1.60E-05	1.94E-12

Table 2: Accuracy of first and second derivatives for D constructed using $\varepsilon = 10^{-14}$ and varying bandlimits c. For each c, the errors are computed for the functions $\sin(ax)$ and $\cos(ax)$ over a large number of values of a in $[0, c]$.

ε	m	n	First derivatives			Second derivatives		
			$\max_{a\in[0,c]} E_{a,1}$	$\max_{a\in[0,c]} E_{a,2}$	$E_{c,3}$	$\max_{a\in[0,c]} E_{a,1}$	$\max_{a\in[0,c]} E_{a,2}$	$E_{c,3}$
1.0E-07	677	658	1.09E-02	4.78E-04	6.76E-07	6.74E+01	2.98E+00	4.21E-06
1.0E-08	680	664	6.26E-04	2.72E-05	3.85E-08	4.66E+00	2.03E-01	2.87E-07
1.0E-09	682	668	1.44E-04	6.32E-06	8.94E-09	1.18E-01	5.09E-02	7.20E-08
1.0E-10	685	672	7.30E-06	3.16E-07	4.46E-10	6.88E-02	2.94E-03	4.16E-09
1.0E-11	688	674	1.62E-06	6.99E-08	9.88E-11	1.63E-02	6.93E-04	9.81E-10
1.0E-12	691	678	7.23E-08	3.12E-09	4.42E-12	8.25E-04	3.50E-05	4.94E-11
1.0E-13	693	680	1.20E-08	5.16E-10	7.30E-13	1.76E-04	7.10E-06	1.00E-11
1.0E-14	696	682	2.95E-09	1.57E-10	2.22E-13	5.68E-05	2.25E-06	2.81E-12

Table 3: Accuracy of first and second derivatives for D constructed using bandlimit $c = 1000$ and varying values of ε.

Table 4: l_2 norms of the rows of the differentiation matrices constructed for first and second derivatives using varying values of c and ε. E_1 denotes the l_2 norm of the first row of the matrix, while E_2 denotes the l_2 norm of the jth row, where j is the largest integer less than or equal to $m/2$.

c	ε	m	First derivatives		Second derivatives	
			E_1	E_2	E_1	E_2
50	1.0E-08	57	6.84E+02	3.42E+01	2.32E+05	1.60E+03
50	1.0E-14	66	1.26E+03	3.84E+01	8.11E+05	2.12E+03
200	1.0E-08	161	2.59E+03	1.20E+02	3.25E+06	1.96E+04
200	1.0E-14	173	4.77E+03	1.26E+02	1.14E+07	2.14E+05
600	1.0E-08	422	7.50E+03	3.51E+02	2.69E+07	1.66E+05
600	1.0E-14	437	1.35E+04	3.56E+02	8.92E+07	1.71E+05
1400	1.0E-08	936	1.85E+04	8.13E+02	1.66E+08	8.90E+05
1400	1.0E-14	954	3.23E+04	8.18E+02	5.15E+08	9.07E+05

In addition to looking at the accuracies of the differentiation matrices on functions of the form $\sin(ax)$ and $\cos(ax)$, we look at the accuracies when the matrices are used to approximate the second derivatives of the functions $\exp(-1000x^2)$, $\sin(20\pi x + \sin(20\pi x))$, and $\cos^2(300x)$ on the interval $[-1, 1]$. Figure 4 shows the relative l_2 errors, with the differentiation matrices constructed using $\varepsilon = 10^{-14}$ and varying values of c in the interval $[0, 1000]$. The errors are computed on the nodes corresponding to the matrices.

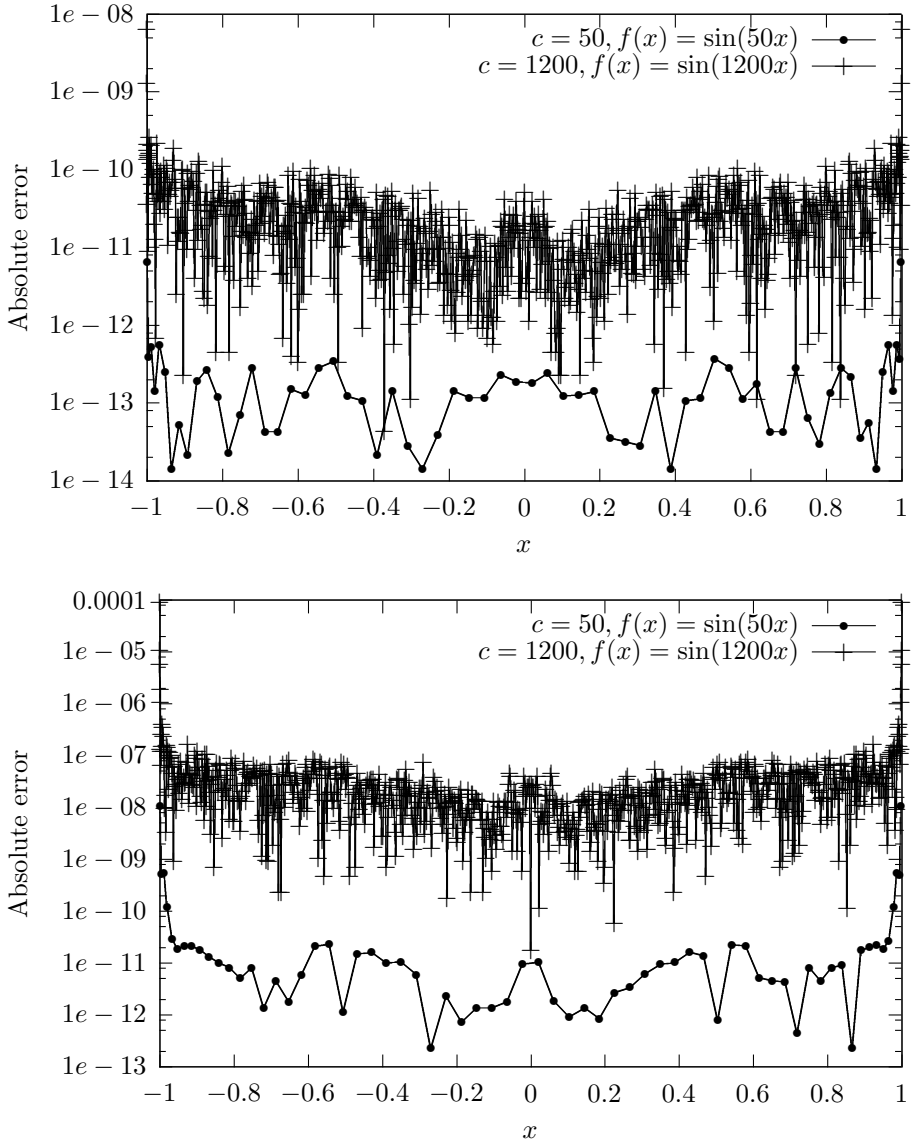

Figure 1: Absolute errors in (a) first and (b) second derivatives of $f(x) = \sin(cx)$ for D constructed using $\varepsilon = 10^{-14}$ and $c = 50, 1200$. The errors are computed at the interpolation nodes on $[-1, 1]$ corresponding to D.

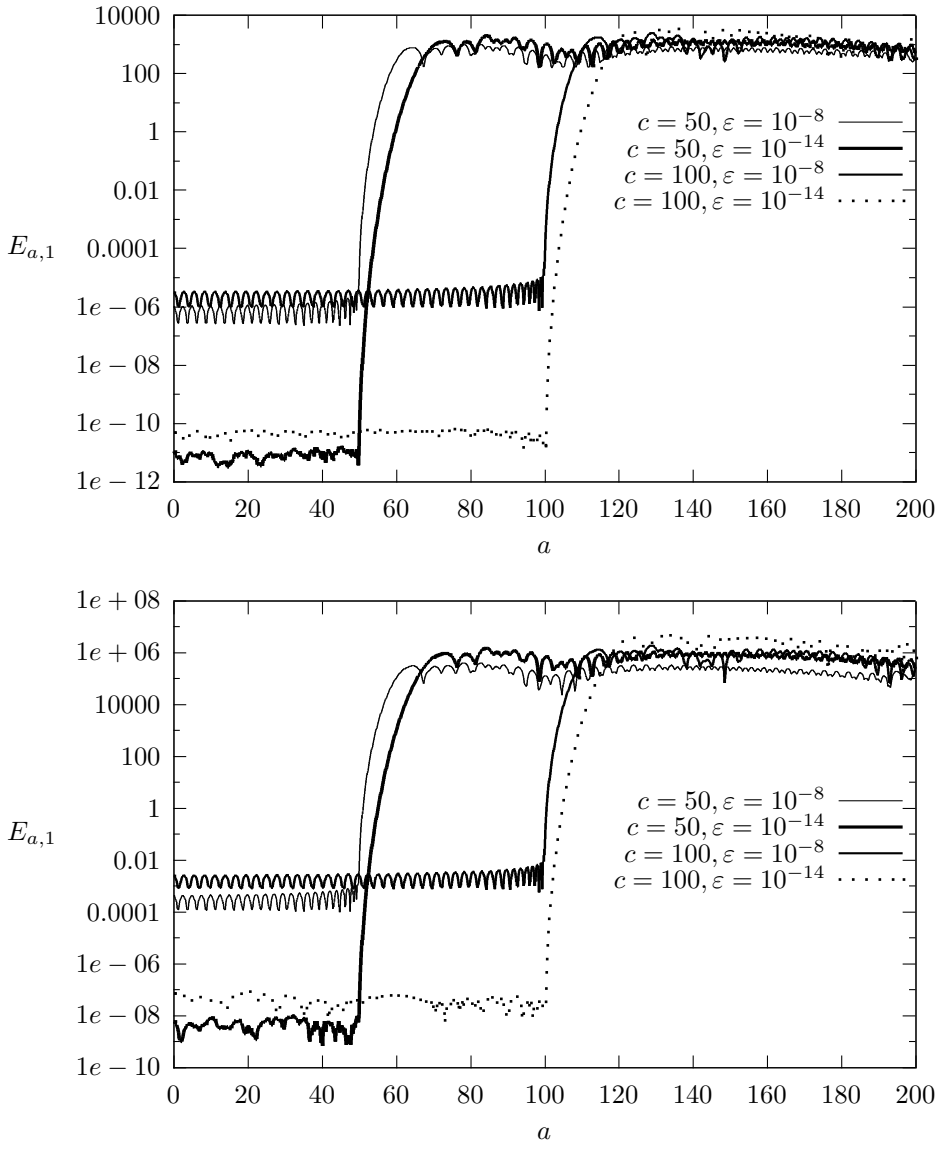

Figure 2: The errors $E_{a,1}$ across $a \in [0, 200]$ for (a) first and (b) second derivatives, where D is constructed using $c = 50, 100$ and $\varepsilon = 10^{-8}, 10^{-14}$.

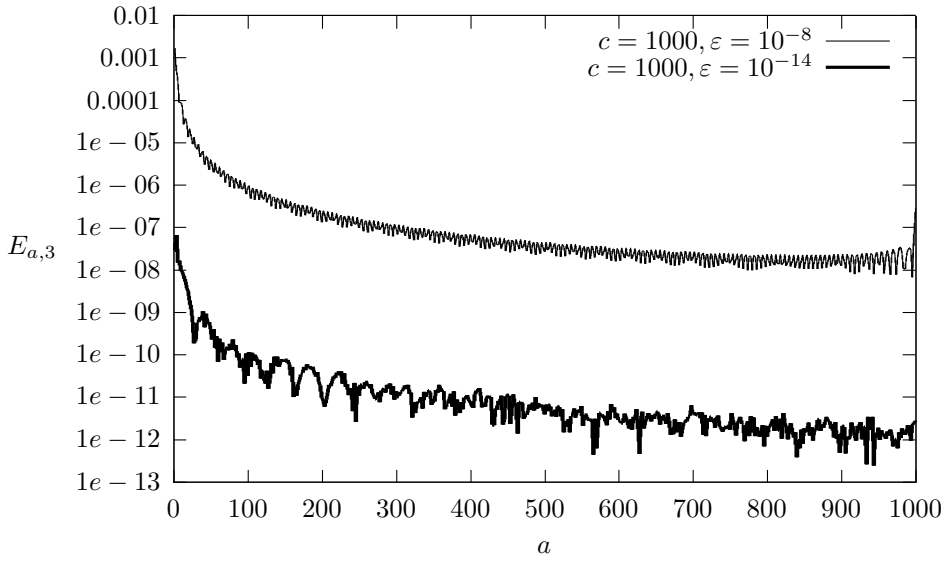

Figure 3: The errors $E_{a,3}$ across $a \in [0, 1000]$ for the second derivative matrix constructed using $c = 1000$ and $\varepsilon = 10^{-8}, 10^{-14}$.

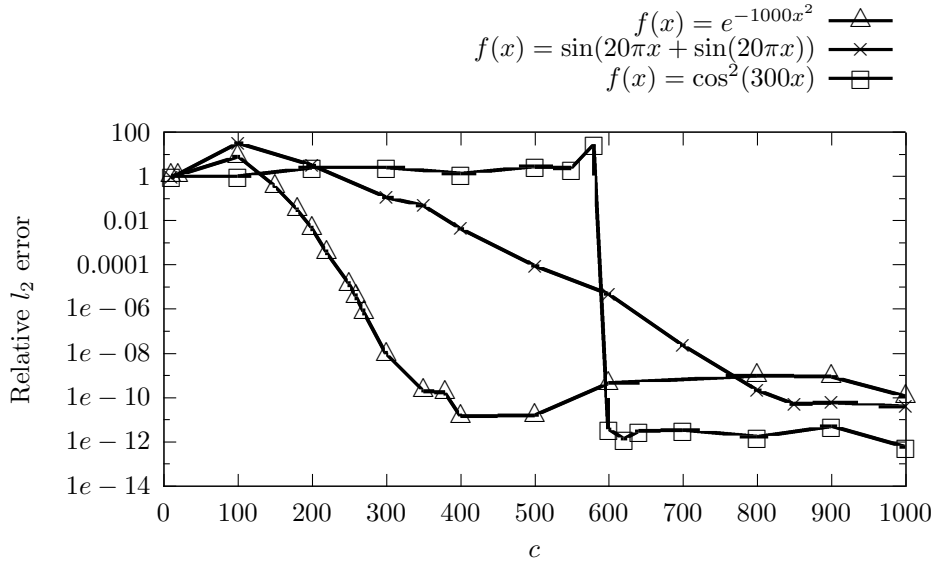

Figure 4: Relative l_2 errors of second derivatives for the functions $f(x) = \exp(-1000x^2)$, $\sin(20\pi x + \sin(20\pi x))$, and $\cos^2(300x)$ defined on $[-1, 1]$. The second derivative matrices are constructed using $\varepsilon = 10^{-14}$ and varying values of c in $[0, 1000]$.

4.1.2 Eigenvalues of the differentiation matrices

One of the main applications of differentiation matrices is the solution of time-dependent PDEs. Often, the spatial derivatives of a time-dependent PDE are discretized by differentiation matrices that incorporate the boundary conditions; the resulting semi-discretized system is then solved using a numerical ODE solver, such as the Runge-Kutta scheme with a suitably chosen time-step Δt. In many practical situations, the stability of such a scheme for the solution of time-dependent PDEs is determined by the eigenvalues of the discretized spatial operator. Namely, the eigenvalues of the discretized spatial operator, multiplied by the time-step Δt, have to lie inside the stability region of the ODE solver in order to ensure stability. In this subsection, we present results related to the eigenvalues of the differentiation matrices constructed using the scheme described in Section 3.2, and compare them to the eigenvalues of the differentiation matrices constructed using the Chebyshev collocation method. Unless otherwise stated, all eigenvalues are computed numerically in double precision arithmetic.

Second-order differentiation matrices

We first consider the wave equation with the homogeneous Dirichlet boundary condition:

$$\begin{aligned} u_{tt} &= u_{xx}, & x &\in (-1,1), \quad t > 0, \\ u(x,0) &= f(x), & x &\in [-1,1], \\ u(-1,t) &= u(1,t) = 0, & t &\geq 0, \end{aligned} \tag{4.4}$$

and construct using the scheme described in Section 3.2 a differentiation matrix D that discretizes the operator u_{xx} and incorporates the boundary condition. The constructed matrix D also discretizes the eigenvalue problem

$$\begin{aligned} u_{xx} &= \lambda u, \ x \in (-1,1), \\ u(\pm 1) &= 0, \end{aligned} \tag{4.5}$$

which has analytical eigenvalues

$$\lambda_j = -\frac{j^2\pi^2}{4}, \ j = 1, 2, \ldots, \tag{4.6}$$

and corresponding eigenfunctions

$$u(x) = \sin\frac{j\pi(x+1)}{2}, \ j = 1, 2, \ldots. \tag{4.7}$$

Table 5 shows the non-zero eigenvalues of the matrix D constructed with $c = 20\pi$ and $\varepsilon = 10^{-13}$. There are 66 non-zero eigenvalues, which is the same as the number of functions k in the orthonormal set $\phi_1, \ldots, \phi_k$ (3.18) corresponding to the chosen c and ε. For ease of comparison with (4.6), the eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{66}$ in Table 5 are scaled by the factor $4/\pi^2$. From the table, we see that $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{66}$ are real, distinct, and negative; and $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{40}$ are accurate to at least 13 digits with respect to the eigenvalues of the continuous problems (4.6), which is expected given the choice of the c used in the construction of

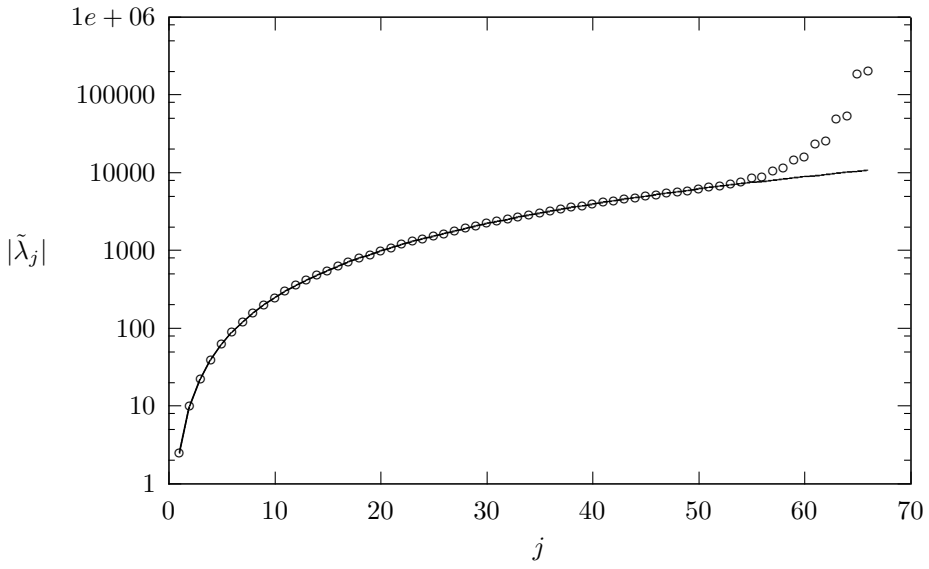

Figure 5: Plot of the absolute values of the eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{66}$ in Table 5 compared with the analytical eigenvalues (4.6) drawn on the solid line.

D. For $k > 40$, the magnitudes of $\tilde{\lambda}_j$ start to grow exponentially, and they no longer approximate the eigenvalues (4.6). Figure 5 shows the plot of $|\tilde{\lambda}_j|$ compared with the eigenvalues (4.6).

In addition, we compute the eigenvalues of the matrix D constructed using $c = 600\pi$ and $\varepsilon = 10^{-13}$. There are 1246 non-zero eigenvalues $\tilde{\lambda}_j$. All of them are again real, distinct, and negative, and agree with (4.6) to at least 13 digits, for all $1 \leq j \leq 1200$. Figure 6 shows the plot of $|\tilde{\lambda}_j|$ compared with (4.6). The plot shows that $|\tilde{\lambda}_j|$ begin to grow exponentially for $j > 1200$.

Remark 4.1. We construct the above differentiation matrices using the scheme described in Section 3.2, in which we use the algorithm of [69] to construct a quadrature integrating all functions on $[-1, 1]$ of bandlimit $2c$ to precision ε^2. Our numerical experiments indicate that such choice of quadrature is sufficient in discretizing the functions $\tilde{\psi}_0, \ldots, \tilde{\psi}_{n-1}$ (3.16). Unless otherwise stated, we will adopt this choice of quadrature for the rest of this paper. As pointed out in Remark 3.7, as long as the choice of c and ε is fixed, increasing the precision of the quadrature beyond ε^2, and hence increasing the size of D, has no effect on the non-zero eigenvalues.

In the numerical solution of a time-dependent PDE using a combination of a differentiation matrix with an explicit time-marching scheme, the largest size of the time-step Δt that maintains stability is often determined by the spectral radius of the differentiation matrix, defined as the maximum moduli of its eigenvalues. In the following, we look at the spectral radii of the differentiation matrices D discretizing (4.5) constructed using the scheme described in Section 3.2, and compare them with those of the differentiation matrices constructed using the Chebyshev collocation method. Let $\hat{D}_C \in \mathbb{R}^{N \times N}$ denote the second derivative matrix constructed using the Chebyshev collocation method on N collocation points, as described in Section 2.3.2. There are several ways of imposing the zero boundary conditions

Table 5: Non-zero eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{66}$ of the differentiation matrix D discretizing the problem (4.5), constructed by the scheme described in Section 3.2 using $c = 20\pi$ and $\varepsilon = 10^{-13}$. Each $\tilde{\lambda}_j$ is scaled by the factor $4/\pi^2$.

-.1000000000000189E+01	-.4000000000000032E+01
-.9000000000000133E+01	-.1600000000000005E+02
-.2500000000000034E+02	-.3600000000000038E+02
-.4900000000000004E+02	-.6400000000000021E+02
-.8100000000000016E+02	-.1000000000000002E+03
-.1210000000000001E+03	-.1440000000000000E+03
-.1690000000000004E+03	-.1960000000000000E+03
-.2250000000000000E+03	-.2559999999999995E+03
-.2889999999999992E+03	-.3240000000000014E+03
-.3610000000000005E+03	-.4000000000000004E+03
-.4409999999999986E+03	-.4840000000000009E+03
-.5290000000000006E+03	-.5760000000000006E+03
-.6250000000000019E+03	-.6759999999999987E+03
-.7289999999999994E+03	-.7840000000000028E+03
-.8409999999999986E+03	-.9000000000000000E+03
-.9609999999999964E+03	-.1024000000000003E+04
-.1089000000000000E+04	-.1156000000000000E+04
-.1224999999999996E+04	-.1296000000000005E+04
-.1369000000000002E+04	-.1444000000000000E+04
-.1520999999999995E+04	-.1600000000000006E+04
-.1681000000000003E+04	-.1764000000000001E+04
-.1849000000000964E+04	-.1936000000030938E+04
-.2025000012389790E+04	-.2116000113341137E+04
-.2209009537738922E+04	-.2304041387127145E+04
-.2401989463349080E+04	-.2502490547117806E+04
-.2602587546324507E+04	-.2737428098562087E+04
-.2930504310944726E+04	-.3083256186436362E+04
-.3435568410972565E+04	-.3645848773066469E+04
-.4308191253136334E+04	-.4612338137669602E+04
-.5951192152853712E+04	-.6426158975995367E+04
-.9541230907887420E+04	-.1038294717510027E+05
-.1984784397165518E+05	-.2173440910500743E+05
-.7558406595445475E+05	-.8310762526925356E+05

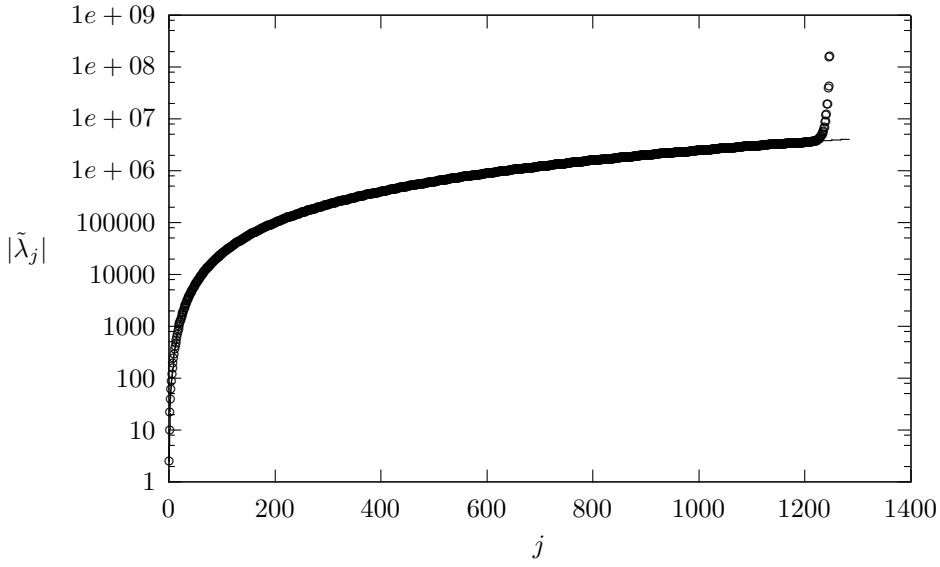

Figure 6: Plot of the absolute values of the non-zero eigenvalues $\tilde{\lambda}_j$ of the differentiation matrix D discretizing the problem (4.5), constructed by the scheme described in Section 3.2 using $c = 600\pi$ and $\varepsilon = 10^{-13}$. There are 1246 of these eigenvalues, and they are compared with the analytical eigenvalues (4.6) drawn on the solid line.

$u(\pm 1) = 0$ on $\hat{D}_C$ (see, for instance, [8, 23, 63]), one of which is to strip $\hat{D}_C$ of its first and last rows and columns, resulting in the $(N-2) \times (N-2)$ matrix D_C. In [29], it is proven that all eigenvalues of D_C are real, distinct, and negative.

Tables 6 and 7 show the spectral radii of D constructed with varying values of c and $\varepsilon = 10^{-7}$ and 10^{-13}, respectively. All computed eigenvalues are either zero, or real, distinct, and negative, and the spectral radii are compared with those of D_C. The following notation is used when presenting the results.

$$
\begin{aligned}
m &= \text{number of nodes used in the construction of } D, \text{ with} \\
&\quad \text{bandlimit parameter chosen as } c \\
N &= \text{number of nodes used in the construction of } D_C, \\
&\quad \text{corresponding to } D_C \text{ of dimension } N-2 \\
E_c &= \text{relative } l_2 \text{ error when } D \text{ or } D_C \text{ is used to approximate} \\
&\quad \text{the second derivatives of the functions } \sin(cx) - x\sin(c) \\
&\quad \text{and } \cos(cx) - \cos(c) \text{ on } [-1,1] \\
\rho &= \text{spectral radius of } D \text{ or } D_C
\end{aligned}
$$

The errors E_c are computed on the quadrature nodes or the Chebyshev collocation points corresponding

to D or D_C respectively; and the functions $\sin(cx) - x\sin(c)$ and $\cos(cx) - \cos(c)$ are just sine and cosine functions with linear functions subtracted to make them satisfy the zero condition at $x = \pm 1$. For each value of c, we choose the number of nodes N for the Chebyshev differentiation matrix D_C to give similar error E_c as does the corresponding D. The number of nodes m for D is determined by the choice of c and ε, as well as the choice of quadrature (see Remark 4.1). In the following, we make several observations and remarks regarding the results.

1. From Tables 6 and 7, we observe that for fixed ε, the spectral radius ρ of D grows like c^2. Figure 7 shows the plot of ρ against c on a log-log scale, which verifies the observation.

2. Since, as pointed out in Section 4.1.1, the ratio of number of nodes m to c/π is roughly equal to 2 for large c, we can interpret the observation of (1) as saying that the spectral radius of D grows roughly as m^2 for large m. However, caution must be taken when interpreting the results this way, since m is not an independent variable in our construction of the matrices.

3. The spectral radius ρ of D_C, on the other hand, grows as N^4, as illustrated in Figure 8. This is a well-known property of collocation methods on both Chebyshev and Legendre points (see, for instance, [68]), and it imposes strict stability condition on the time-step allowed in an explicit time-stepping scheme.

   For example, from Table 7, we see that the differentiation matrix D constructed with $c = 2000$ and $\varepsilon = 10^{-13}$ has spectral radius 1.53E+08, while the Chebyshev matrix D_C that yields similar error E_c has spectral radius 9.19E+11. This means that for the solution of the parabolic equation $u_t = u_{xx}$, the step-size required to maintain stability when using D_C to discretize u_{xx} are about 6000 times smaller than those needed when using D to discretize u_{xx}. For the solution of the hyperbolic equation $u_{tt} = u_{xx}$, the ratio of the required step-sizes is about $\sqrt{6000} \approx 77$. The numerical results in Section 4.2 further illustrate the difference in stability requirements implied by the spectral radii corresponding to the two differentiation schemes.

4. From Tables 6 and 7, we see that for functions with large bandlimits, the number of nodes required by the Chebyshev collocation method is about $\pi/2$ times more than the nodes required by the scheme described in Section 3.2. The results are in agreement with Theorem 2 of [68], which asserts that asymptotically as the bandwidth a increases, about π points per wavelength are needed for the interpolation of the functions $\sin(ax)$ and $\cos(ax)$ on the Chebyshev collocation points.

Table 6: Spectral radii ρ of the differentiation matrices D discretizing the problem (4.5), constructed by the scheme described in Section 3.2, with $\varepsilon = 10^{-7}$ and varying values of c. For each c, the spectral radius is compared with that of the Chebyshev matrix D_C that gives comparable error E_c.

		PSWFs			Chebyshev	
c	m	E_c	ρ	N	E_c	ρ
5	13	2.25E-05	5.63E+02	15	8.56E-06	1.87E+03
10	18	1.94E-06	2.29E+03	23	2.13E-06	1.12E+04
20	26	2.61E-06	6.93E+03	36	1.66E-06	7.14E+04
40	41	6.20E-06	2.13E+04	58	7.16E-06	5.01E+05
50	48	1.95E-06	3.77E+04	71	1.65E-06	1.14E+06
80	68	3.48E-06	8.47E+04	104	2.02E-06	5.33E+06
100	82	1.94E-06	1.40E+05	126	1.68E-06	1.16E+07
200	150	1.72E-06	5.42E+05	233	1.22E-06	1.37E+08
400	280	1.21E-06	2.19E+06	441	1.42E-06	1.78E+09
500	344	1.62E-06	3.28E+06	545	9.81E-07	4.15E+09
800	534	2.40E-06	7.94E+06	850	2.44E-06	2.46E+10
1000	665	2.34E-06	1.23E+07	1055	1.67E-06	5.85E+10
1200	793	2.18E-06	1.77E+07	1258	1.92E-06	1.18E+11
1600	1049	1.65E-06	3.25E+07	1664	1.84E-06	3.62E+11
2000	1302	1.19E-06	5.30E+07	2070	1.40E-06	8.68E+11

Table 7: Spectral radii ρ of the differentiation matrices D discretizing the problem (4.5), constructed by the scheme described in Section 3.2, with $\varepsilon = 10^{-13}$ and varying values of c.

		PSWFs			Chebyshev	
c	m	E_c	ρ	N	E_c	ρ
5	18	3.77E-11	2.82E+03	23	2.33E-12	1.12E+04
10	24	1.942-11	7.84E+03	32	1.33E-12	4.40E+04
20	34	6.26E-12	2.55E+04	46	4.01E-12	1.95E+05
40	50	8.39E-12	8.18E+04	71	1.21E-11	1.14E+06
50	59	4.53E-12	1.28E+05	84	5.49E-12	2.25E+06
80	81	3.26E-12	3.12E+05	120	3.46E-12	9.50E+06
100	95	3.09E-12	4.74E+05	143	3.09E-12	1.93E+07
200	161	1.98E-11	1.57E+06	250	1.87E-11	1.82E+08
400	292	1.12E-11	6.38E+06	467	1.65E-11	2.23E+09
500	360	4.37E-12	1.05E+07	575	3.70E-11	5.14E+09
800	554	3.68E-12	2.68E+07	890	4.11E-11	2.96E+10
1000	682	6.61E-12	4.02E+07	1080	2.38E-10	6.42E+10
1200	811	1.00E-11	5.64E+07	1290	1.13E-10	1.31E+11
1600	1067	1.24E-11	9.81E+07	1690	5.67E-10	3.85E+11
2000	1324	1.05E-11	1.53E+08	2100	4.37E-10	9.19E+11

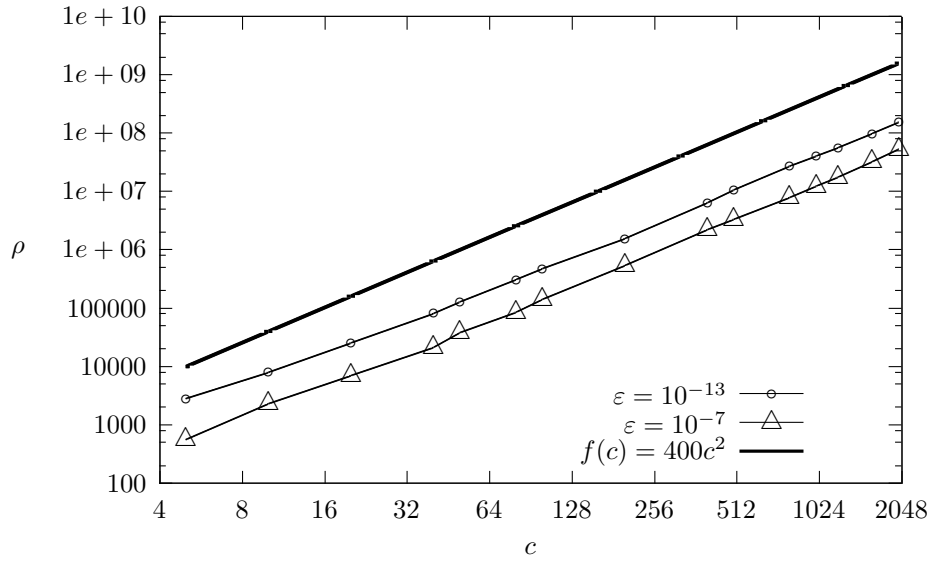

Figure 7: Log-log plot of the spectral radius ρ of the differentiation matrix D shown in Tables 6 and 7 against c.

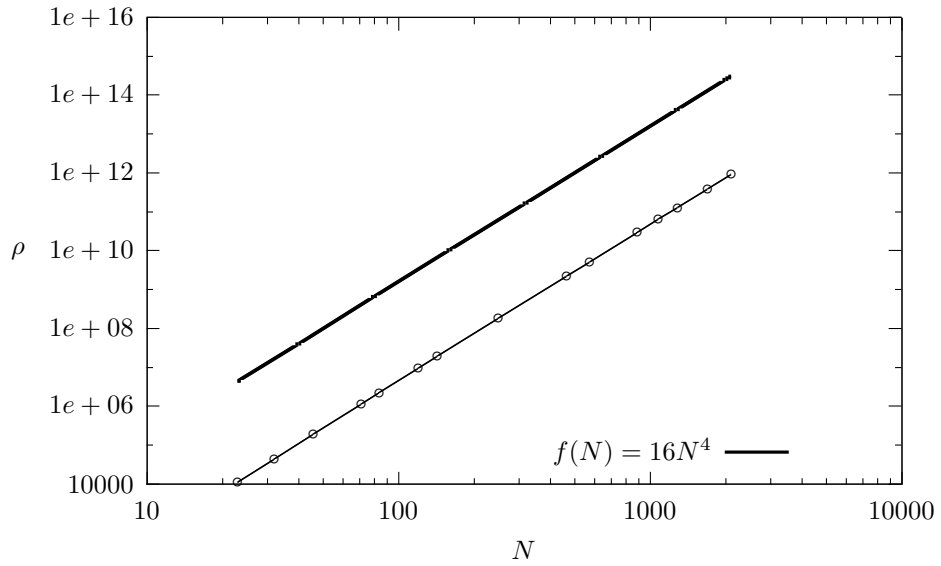

Figure 8: Log-log plot of the spectral radius ρ of the Chebyshev differentiation matrix D_C shown in Table 7 against N.

First-order differentiation matrices

In the remaining part of this subsection, we look at the eigenvalues of the first derivative matrices constructed by the scheme described in Section 3.2, and compare them to those of the matrices constructed using the Chebyshev collocation method. We consider the first-order hyperbolic initial boundary value problem

$$\begin{aligned} u_t &= u_x, & x &\in (-1,1), \quad t > 0, \\ u(x,0) &= f(x), & x &\in [-1,1], \\ u(1,t) &= 0, & t &\geq 0, \end{aligned} \tag{4.8}$$

and discretize the spatial derivative operator u_x by the obvious modification of the scheme described in Section 3.2. Namely, we change the μ_i defined in (3.15) into the constant functions

$$\mu_i(x) = \psi_i(1), \tag{4.9}$$

and change the second derivatives in (3.24) and (3.25) into first derivatives. We denote the resulting differentiation matrix by D.

Remark 4.2. The other modification is that here we are subtracting from the PSWFS $\psi_0, \ldots, \psi_{n-1}$ one degree of freedom, so the matrix A defined in (3.17) has ε-rank equal to $n-1$, where ε is the precision with respect to which $\psi_1, \ldots, \psi_n$ are chosen.

Figure 9 shows on the complex plane the eigenvalues of the matrix D constructed using $c = 20\pi$ and $\varepsilon = 10^{-13}$. There are 67 non-zero eigenvalues, which equals the number of functions k in the orthonormal set $\phi_1, \ldots, \phi_k$ (3.18) corresponding to c and ε. All eigenvalues of D lie on the left half-plane, so stability is ensured when we discretize u_x in (4.8) by D and solve the resulting system by an explicit time-stepping scheme with sufficiently small time-step. Most of the eigenvalues are distributed along a bow-shaped arc around the origin, with a few outliers extending beyond the arc. The farthest outlying eigenvalues $\lambda_\pm = (-0.407, \pm 226)$ almost touch the imaginary axis, so the spectral radius of D is roughly equal to the magnitude of their imaginary part. Table 8 shows the spectral radii of D constructed using varying values of c and $\varepsilon = 10^{-13}$, and compares them with the spectral radii of the matrices D_C constructed using the Chebyshev collocation method. More specifically, D_C is obtained by constructing the first derivative matrix $\hat{D}_C$ as described in Section 2.3.2, and then removing its first row and column. The

following notation is used when presenting the results.

$$
\begin{aligned}
m &= \text{number of nodes used in the construction of } D, \text{ with} \\
&\quad \text{bandlimit parameter chosen as } c \\
N &= \text{number of nodes used in the construction of } D_C, \\
&\quad \text{corresponding to } D_C \text{ of dimension } N-1 \\
E_c &= \text{relative } l_2 \text{ error when } D \text{ or } D_C \text{ is used to approximate} \\
&\quad \text{the first derivatives of the functions } \sin(cx)-\sin(c) \\
&\quad \text{and } \cos(cx)-\cos(c) \text{ on } [-1,1] \\
\rho &= \text{spectral radius of } D \text{ or } D_C
\end{aligned}
$$

For each value of c, we choose the number of nodes N for D_C to give similar error E_c as does the corresponding D. In addition, we plot in Figures 10 and 11 the eigenvalues of D and D_C corresponding to several selected results in Table 8. In the following, we make several observations and remarks based on the presented results, and on the results of our more extensive experiments.

1. In contrast to the second-order problem (4.4), the first derivative operator u_x in (4.8), with the boundary condition $u(1) = 0$, does not correspond to any eigenvalue problem. Therefore, the eigenvalues of D and D_C are not approximations to any physically meaningful eigenvalues, and are entirely numerical in origin.

2. All eigenvalues of D and D_C lie on the left half-plane. Most of the eigenvalues of D are distributed along a bow-shaped arc, with some outlying eigenvalues extending along the imaginary axis (see Figure 10). As we fix ε and increase c, the imaginary parts of the outlying eigenvalues increase proportionately, while the real parts of all eigenvalues remain bounded. In particular, all of the eigenvalues of D corresponding to the results in Table 8 have real parts bounded by 20. On the other hand, as observed in [64], most of the eigenvalues of D_C are distributed along an arc or a loop, again with some outliers (see Figure 11). In contrast to D, both the real and imaginary parts of the outlying eigenvalues grow as we increase N.

3. For fixed ε, the spectral radius of D grows as c. Again, this can be interpreted as that the spectral radius of D grows roughly as m for large m, with the same precaution as in the second derivative case. On the other hand, the spectral radius of D_C grows as N^2, which is again a well-known property (see, for instance, [19, 64, 68]). As an example, Figures 12 and 13 illustrate for the results in Table 8 the $O(c)$ and $O(N^2)$ growth of the spectral radii of D and D_C, respectively. The difference in the asymptotic behaviors of their spectra leads to different stability requirements in the solution of time-dependent PDEs, which we will investigate in Section 4.2.

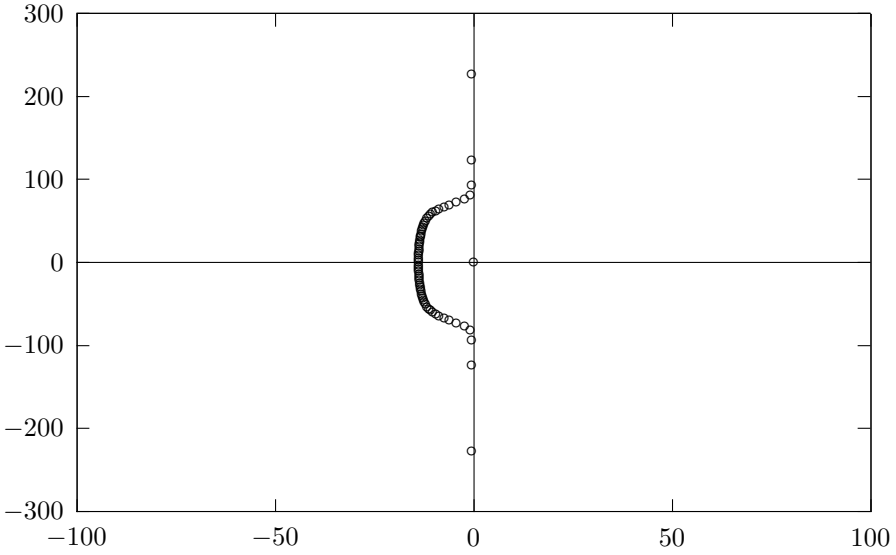

Figure 9: Eigenvalues of the first derivative matrix D incorporating the boundary condition $u(1) = 0$, constructed by the scheme described in Section 3.2 using $c = 20\pi$ and $\varepsilon = 10^{-13}$.

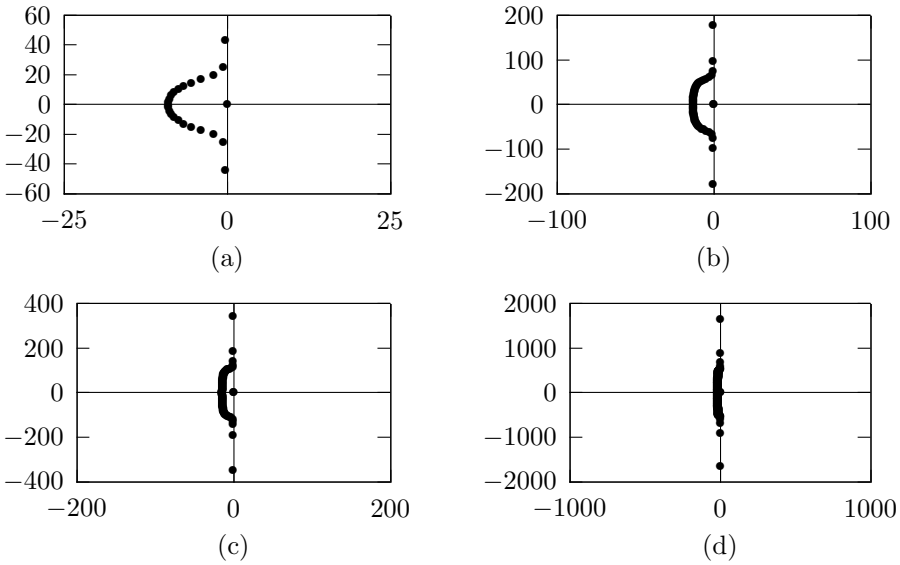

Figure 10: Spectra of D constructed with $\varepsilon = 10^{-13}$ and (from (a) to (d)) $c = 10, 50, 100, 500$.

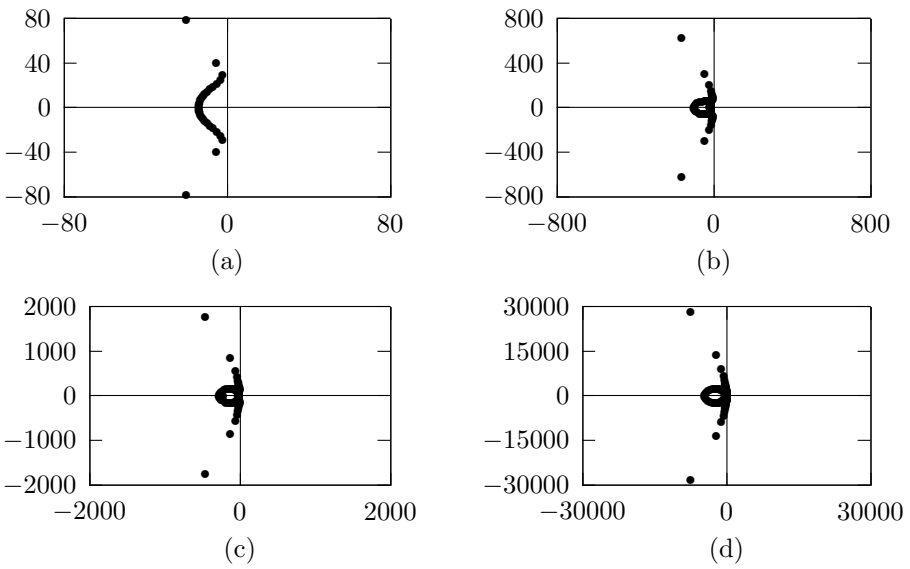

Figure 11: Spectra of D_C constructed with number of nodes (from (a) to (d)) $N = 31, 86, 144, 575$.

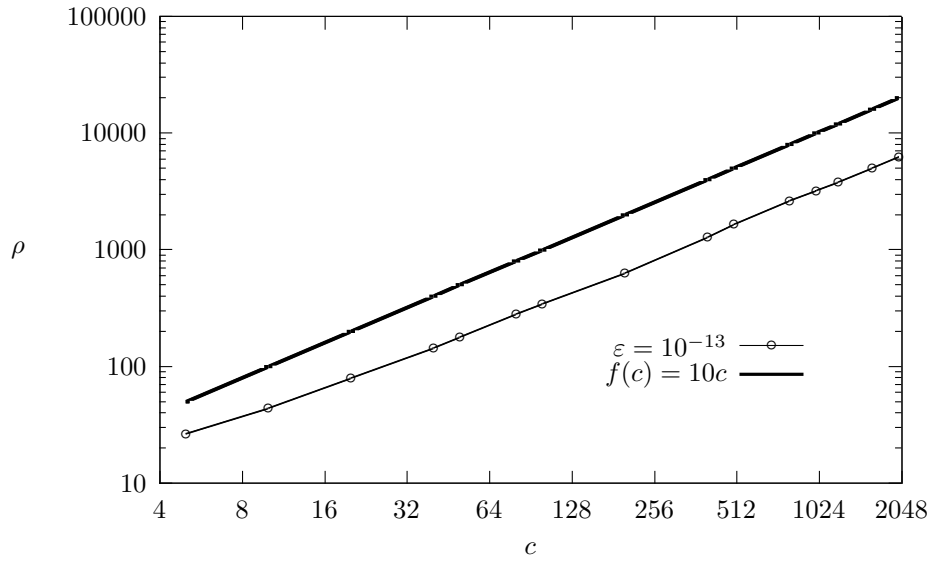

Figure 12: Log-log plot of the spectral radius ρ of the first derivative matrix D shown in Table 8 against c.

Table 8: Spectral radii ρ of D constructed with $\varepsilon = 10^{-13}$ and varying values of c. For each c, the spectral radius is compared with that of the Chebyshev matrix D_C that gives comparable error E_c.

	PSWFs			Chebyshev		
c	m	E_c	ρ	N	E_c	ρ
5	18	2.07E-12	2.63E+01	23	7.92E-13	4.37E+01
10	24	1.27E-12	4.38E+01	31	3.23E-12	8.06E+01
20	34	4.45E-13	7.92E+01	47	4.37E-13	1.88E+02
40	50	7.15E-13	1.43E+02	73	5.61E-13	4.60E+02
50	59	3.91E-13	1.79E+02	86	2.60E-13	6.41E+02
80	81	2.88E-13	2.80E+02	121	4.74E-13	1.28E+03
100	95	2.82E-13	3.45E+02	144	4.51E-13	1.81E+03
200	161	1.93E-12	6.23E+02	253	1.56E-12	5.63E+03
400	292	1.06E-12	1.28E+03	467	9.41E-13	1.92E+04
500	360	3.93E-13	1.64E+03	575	4.00E-13	2.92E+04
800	554	3.43E-13	2.62E+03	890	5.17E-13	7.00E+04
1000	682	6.23E-13	3.21E+03	1090	1.32E-12	1.05E+05
1200	811	8.98E-13	3.81E+03	1300	1.05E-12	1.50E+05
1600	1067	1.14E-12	5.02E+03	1710	1.30E-12	2.59E+05
2000	1324	1.05E-12	6.27E+03	2120	1.52E-12	3.98E+05

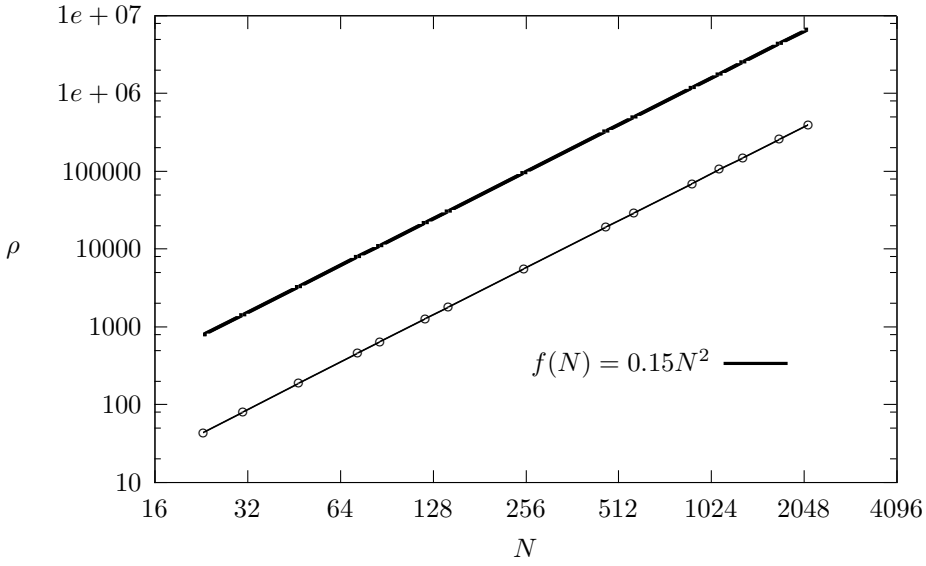

Figure 13: Log-log plot of the spectral radius ρ of the Chebyshev differentiation matrix D_C shown in Table 8 against N.

4.2 Numerical examples

In this subsection, we present the results of several numerical experiments performed to assess the performance of the scheme described in Section 3.2 in the solution of time-dependent partial differential equations, and the associated eigenvalue problems. For the solution of time-dependent PDEs, we discretize the spatial derivatives using the differentiation scheme described in Section 3.2, and solve the resulting system in the time direction with the explicit predictor-corrector schemes PC1 and PC3 described in [26], where PC1 is a 22-step solver that produces 7-digit precision at 22 steps per wavelength, and PC3 is a 42-step solver that produces 15-digit precision at 36 steps per wavelength. The predictor-corrector schemes PC1 and PC3 are implemented with two correction steps per time step, where the time steps are obtained from an equidistant discretization of the given time interval. The starting values for the schemes PC1 and PC3 are computed via the spectral deferred correction schemes DC1 and DC3 of [26], respectively. We refer the readers to [26] for a detailed discussion of the construction of the explicit schemes DC1, DC3, PC1, and PC3, as well as their accuracy and stability properties.

As the benchmark, we compare the differentiation scheme described in Section 3.2 with the Chebyshev collocation method and the fourth-order finite difference scheme (FD4). We assess the performance of both methods in the solution of time-dependent PDEs by combining them with the time-marching schemes PC1, PC3, and also the fourth-order Runga-Kutta scheme (RK4).

All schemes were implemented in FORTRAN 77 and compiled with the Lahey-Fujitsu FORTRAN 95 compiler. All experiments were run on a PC with an Intel Core i7 2.67 GHz processor and 12GB of memory, with only one core utilized. No attempt was made to parallelize any of the code. All computations were performed in double (16-digit) precision.

4.2.1 The wave equation

In this subsection, we consider the following wave equation with homogeneous Dirichlet boundary condition:

$$u_{tt} = u_{xx}, \tag{4.10}$$

$$u(x,0) = \sin(99.5\pi x), u_t(x,0) = 0, \tag{4.11}$$

$$u(0,t) = u(2,t) = 0, \tag{4.12}$$

where $x \in [0,2]$ and $t \geq 0$. The analytical solution of (4.10)-(4.12) is

$$u(x,t) = \sin(99.5\pi x)\cos(99.5\pi t). \tag{4.13}$$

By introducing the auxiliary variable $v = u_t$, we convert (4.10) to a system of two coupled equations

$$\begin{pmatrix} u \\ v \end{pmatrix}_t = \begin{pmatrix} v \\ u_{xx} \end{pmatrix}. \tag{4.14}$$

We then solve the system (4.14), (4.11)-(4.12) numerically on the time interval $[0, 2.5]$ by combining the differentiation scheme described in Section 3.2 with the time-marching schemes PC1 and PC3. We label the combined schemes Prolate+PC1 and Prolate+PC3 respectively. In both of these schemes, the differentiation matrix was constructed with $c = 315$, which is slightly higher than 99.5π. For comparison, we also solve the system (4.14), (4.11)-(4.12) using a combination of the Chebyshev collocation method with the PC1, PC3, and RK4 schemes (which we label Chebyshev+PC1, Chebyshev+PC3, and Chebyshev+RK4 respectively), and a combination of the FD4 and RK4 schemes (which we label FD4+RK4).

Remark 4.3. In all of the above combined schemes, we discretize the operator on the right-hand side of (4.14) by the matrix

$$\begin{pmatrix} 0 & I \\ D & 0 \end{pmatrix}, \tag{4.15}$$

where D is the second derivative matrix incorporating the boundary condition $u(0) = u(2) = 0$, constructed using the corresponding differentiation schemes, and I is an identity matrix with the same dimension as that of D. For both the Chebyshev collocation method and the FD4 scheme, we incorporate the boundary condition $u(0) = u(2) = 0$ by first constructing the second derivative matrix as described in Section 2.3, and then removing its first and last rows and columns. Clearly, in the actual implementations, we only construct D, and never construct or apply (4.15) explicitly.

Tables 9-14 summarize the results for the schemes Prolate+PC1, Prolate+PC3, Chebyshev+PC1, Chebyshev+PC3, Chebyshev+RK4, and FD4+RK4 respectively. In Tables 9 and 10, c and ε denote the bandlimit and precision parameters used in the differentiation scheme described in Section 3.2. In all of the tables, m denotes the number of nodes on the interval $[0, 2]$ used in the construction of the differentiation matrix D. In particular, it corresponds to D of dimension $m \times m$ for the scheme described in Section 3.2, and to D of dimension $(m-2) \times (m-2)$ for the Chebyshev collocation method and the FD4 scheme. ρ denotes the spectral radius of D; n_s denotes the number of nodes in the equidistant discretization of the time interval $[0, 2.5]$, while h denotes the corresponding step-size; E_{l_2} denotes the relative l_2 error of the numerical solution of u at the final time $t_f = 2.5$, computed at the spatial discretization nodes on $[0, 2]$. Finally, T_c denotes the CPU times (in seconds) taken by the scheme described in Section 3.2 to construct the differentiation matrix, and T denotes the CPU time (in seconds) taken by the time-marching scheme.

In addition, we show in Figures 14 and 15 a comparison of the performance of the above schemes in the solution of the system (4.14), (4.11)-(4.12). Figure 14 shows the CPU time T versus the relative error E_{l_2}, and Figure 15 shows the number of time steps n_s versus the relative error E_{l_2}. All data points in Figures 14 and 15 are selected from the results in Tables 9-14.

In the following, we further explain our results, and make several observations and comments.

1. For each set of results in the tables, we first fix the parameters for the differentiation scheme, and then increase the number of time steps n_s until stability is attained and that all observed error is due to the differentiation scheme.

2. For the scheme FD4+RK4, we do not explicitly construct the differentiation matrix D in (4.15). Rather, we only store the differentiation formula (one for differentiation at the first and last interior node on $[0, 2]$, and one for all other interior nodes) and apply them one by one during actual computations. As a result, applying the operator (4.15) involves only $O(m)$ operations. The differentiation matrices constructed by Chebyshev collocation and the scheme described in Section 3.2 were, on the other hand, applied by brute-force multiplications, and no attempt was made to speed up their applications.

3. For the scheme FD4, the differentiation matrix D in (4.15) has real and negative eigenvalues, and its spectral radius grows as m^2 (see, for instance, [8, 15]). The spectral radii ρ shown in Table 14 are in brackets because they are estimated by extrapolation.

4. For the combined schemes that involve either PC1 and PC3 (see Tables 9-12), the number of steps n_s is always dominated by the stability requirement. In other words, once stability is achieved, increasing n_s does not increase accuracy. From [26], the stability regions of the schemes PC1 and PC3 are subsets of the closed left half-plane. We note that the eigenvalues of the matrix (4.15) are purely imaginary, because they are complex square roots of the eigenvalues of the differentiation matrix D, and the latter eigenvalues are all real and negative (see Section 4.1.2). Therefore, the stability requirement on the step-size h is easily determined from the span of the stability regions over the imaginary axis and the spectral radii ρ of D shown in the tables. In particular, the scheme PC1 becomes stable when $\lambda h < 0.55$, while the scheme PC3 becomes stable when $\lambda h < 0.33$, where λ is the maximum imaginary part of the eigenvalues of the matrix (4.15).

5. When compared to the schemes Chebyshev+PC1, Chebyshev+PC3, and

   Chebyshev+RK4, the schemes Prolate+PC1 and Prolate+PC3 consistently achieve speed-up across all accuracy levels. The speed-up comes from smaller dimensions of differentiation matrices, and from smaller number of steps n_s needed to maintain stability. For instance, from Tables 10 and 12, the CPU time T it takes for the scheme Prolate+PC3 to achieve an accuracy of 10^{-12} is about 36 times less than it takes for the scheme Chebyshev+PC3. If the CPU time T_c taken in the construction of the differentiation matrix is included, the scheme Prolate+PC3 still demonstrates a speed-up of 28 times compared to the scheme Chebyshev+PC3.

6. The time interval [0,2.5] contains about 125 periods of the solution (4.13), which means it takes about 40 and 120 steps per period for the schemes Prolate+PC1 and Prolate+PC3 to achieve an accuracy of 10^{-6} and 10^{-12}, respectively. Again, these numbers of steps per periods are higher than those specified by the schemes PC1 and PC3 (see [26]) because of the stability restriction imposed by the differentiation scheme.

7. From Tables 9 and 10, the CPU times T_c taken by the scheme of Section 3.2 to construct the differentiation matrices vary little with the precision ε, and they are comparable in magnitude to the marching times T. It should be kept in mind, however, that the times T_c are fixed given c and ε, i.e., they are independent on the length of the time interval t_f on which marching is performed.

The main components of T_c come from the construction of the interpolatory quadrature using the algorithm of [69], and from the evaluation of the PSWFs. While there are ways to reduce the cost of both components (in terms of asymptotic CPU requirements and associated proportionality constants), we have made no effort to do so in our implementations.

Table 9: Performance of the scheme Prolate+PC1 in the solution of (4.14), (4.11)-(4.12).

c	ε	m	ρ	n_s	h	E_{l_2}	T_c	T
315	10^{-5}	219	7.90E+05	4050	6.01E-04	1.04E-05	1.14	0.88
	10^{-6}	222	9.94E+05	4550	5.50E-04	2.39E-06	1.14	0.98
	10^{-7}	223	1.23E+06	5050	4.95E-04	9.13E-07	1.12	1.12
	10^{-8}	225	1.50E+06	6060	4.13E-04	1.60E-06	1.17	1.26

Table 10: Performance of the scheme Prolate+PC3 in the solution of (4.14), (4.11)-(4.12).

c	ε	m	ρ	n_s	h	E_{l_2}	T_c	T
315	10^{-5}	219	7.90E+05	6750	3.70E-04	1.04E-05	1.14	1.72
	10^{-6}	222	9.94E+05	7550	3.36E-04	2.33E-06	1.14	1.94
	10^{-7}	223	1.23E+06	8400	2.98E-04	6.26E-07	1.12	2.20
	10^{-8}	225	1.50E+06	9300	2.69E-04	7.24E-08	1.17	2.50
	10^{-9}	228	2.14E+06	11 100	2.25E-04	4.28E-10	1.18	3.01
	10^{-10}	230	2.51E+06	12 000	2.08E-04	3.07E-10	1.20	3.24
	10^{-11}	232	2.92E+06	13 000	1.92E-04	6.18E-11	1.22	3.71
	10^{-12}	234	3.37E+06	14 000	1.79E-04	9.53E-12	1.26	4.01
	10^{-13}	238	3.86E+06	15 000	1.67E-04	1.73E-12	1.29	4.31

Table 11: Performance of the scheme Chebyshev+PC1 in the solution of (4.14), (4.11)-(4.12).

m	ρ	n_s	h	E_{l_2}	T
330	5.55E+08	108 000	2.31E-05	1.28E-03	52.03
330	5.89E+08	111 000	2.25E-05	7.43E-05	55.42
340	6.26E+08	114 000	2.19E-05	2.61E-05	57.67
345	6.63E+08	117 500	2.13E-05	2.48E-05	62.04

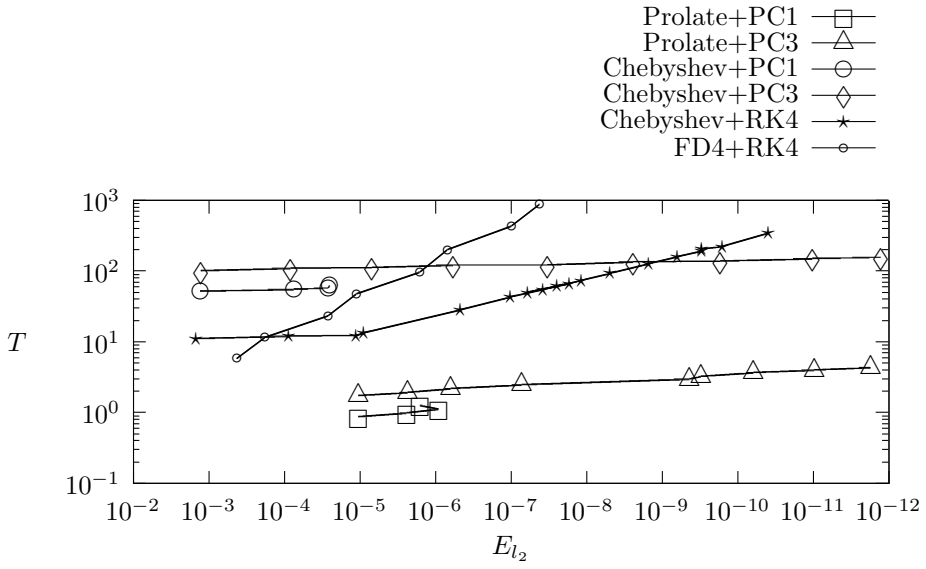

Figure 14: CPU time T versus relative error E_{l_2} in the solution of (4.14), (4.11)-(4.12).

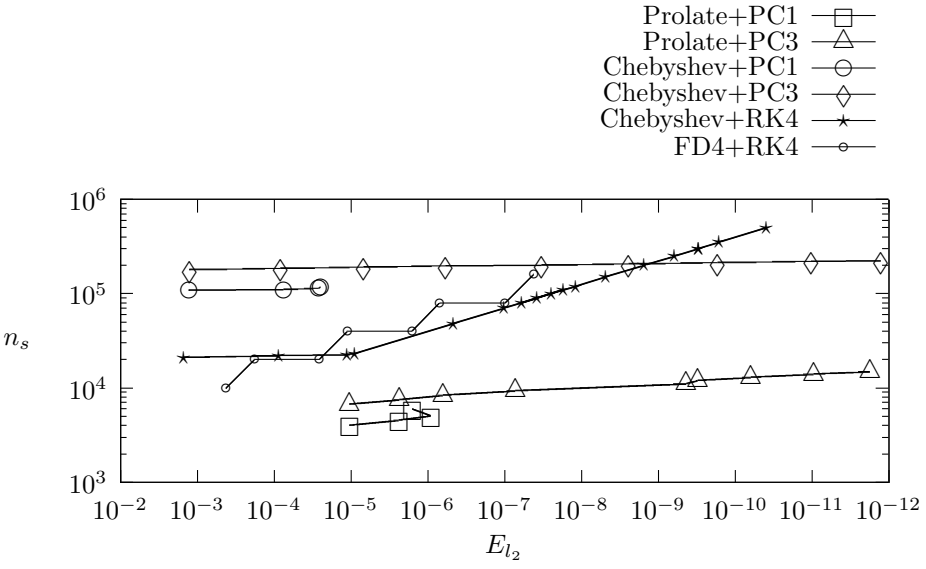

Figure 15: Number of time steps n_s versus relative error E_{l_2} in the solution of (4.14), (4.11)-(4.12).

Table 12: Performance of the scheme Chebyshev+PC3 in the solution of (4.14), (4.11)-(4.12).

m	ρ	n_s	h	E_{l_2}	T
330	5.55E+08	178 500	2.31E-05	1.26E-03	101.65
335	5.89E+08	184 000	1.36E-05	8.24E-05	108.87
340	6.26E+08	189 500	1.32E-05	6.85E-06	111.36
345	6.63E+08	195 500	1.28E-05	5.81E-07	121.38
350	7.03E+08	201 000	1.24E-05	3.31E-08	123.12
355	7.44E+08	207 000	1.21E-05	2.41E-09	135.98
360	7.87E+08	212 500	1.18E-05	1.68E-10	137.81
365	8.32E+08	218 500	1.14E-05	1.02E-11	149.42
368	8.59E+08	222 100	1.13E-05	1.27E-12	156.92

Table 13: Performance of the scheme Chebyshev+RK4 in the solution of (4.14), (4.11)-(4.12).

m	ρ	n_s	h	E_{l_2}	T
330	5.55E+08	21 100	1.12E-04	1.49E-03	10.99
335	5.89E+08	22 000	1.14E-04	8.79E-05	11.96
340	6.26E+08	22 400	1.11E-04	1.12E-05	12.34
345	6.63E+08	23 000	1.09E-05	8.96E-06	13.32
		46 000	5.43E-06	8.10E-07	26.94
350	7.03E+08	24 000	1.04E-04	7.52E-06	14.05
		48 000	5.21E-05	4.65E-07	28.18
		60 000	4.17E-05	1.92E-07	35.06
355	7.44E+08	70 000	3.57E-05	1.02E-07	42.80
		90 000	2.78E-05	3.76E-08	55.01
		110 000	2.27E-05	1.71E-08	66.97
360	7.87E+08	80 000	3.13E-05	5.98E-08	49.65
		100 000	2.50E-05	2.45E-08	61.49
		120 000	2.08E-05	1.18E-08	73.70
		150 000	1.67E-05	4.85E-09	92.08
365	8.32E+08	200 000	1.25E-05	1.52E-09	127.69
		250 000	1.00E-05	6.22E-10	159.59
		300 000	8.33E-06	2.99E-10	193.50
		350 000	7.11E-06	1.61E-10	224.30
368	8.59E+08	300 000	8.33E-05	3.01E-10	205.63
		500 000	5.00E-06	3.90E-11	343.58

Table 14: Performance of the scheme FD4+RK4 in the solution of (4.14), (4.11)-(4.12).

m	ρ	n_s	h	E_{l_2}	T
8000	(1.14E+08)	10 000	2.25E-04	4.20E-04	5.85
		20 000	1.25E-04	1.78E-04	11.64
16 000	(4.54E+08)	20 000	1.25E-04	2.58E-05	23.68
		40 000	6.25E-04	1.11E-05	47.24
32 000	(1.81E+09)	40 000	6.25E-04	1.60E-06	97.43
		80 000	3.13E-04	6.92E-07	200.82
64 000	(7.27E+09)	80 000	3.13E-04	9.74E-08	435.42
		160 000	1.56E-04	4.13E-08	879.46

In the rest of this subsection, we consider the following wave equation with homogeneous Dirichlet boundary condition:

$$u_{tt} = u_{xx}, \tag{4.16}$$

$$u(x,0) = \sin\left(\left(a+\frac{1}{2}\right)\pi x\right), u_t(x,0) = 0, \tag{4.17}$$

$$u(0,t) = u(2,t) = 0, \tag{4.18}$$

where $x \in [0,2]$, $t \geq 0$, and a is a positive integer, of which analytical solution is

$$u(x,t) = \sin\left(\left(a+\frac{1}{2}\right)\pi x\right)\cos\left(\left(a+\frac{1}{2}\right)\pi t\right). \tag{4.19}$$

The value $a + 1/2$ is the wavenumber of the solution u on the spatial interval $[0,2]$.

We present in Table 15 the performance of the scheme Prolate+PC3 in the solution of (4.16)-(4.18) for a range of values of a. For each a, we solve (4.16)-(4.18) on the time interval $[0,t_f]$, where t_f is chosen such that the analytical solution (4.19) spans about 125 periods on $[0,t_f]$, and compute the relative l_2 error E_{l_2} of the numerical solution at time t_f. The parameters for the scheme Prolate+PC3 are chosen such that about 12 digits of accuracy is obtained. Tables 16 and 17 list for the schemes Chebyshev+PC3 and Chebyshev+RK4 the corresponding parameters and timings to obtain similar accuracies. The notation in Tables 15-17 are the same as that in Tables 9-14.

In the following, we make several observations and remarks regarding the results.

1. Again, for the schemes Prolate+PC3 and Chebyshev+PC3, the number of time steps n_s is dominated by the stability requirement. For a fixed precision ε, the differentiation matrix D in (4.15) constructed by the scheme described in Section 3.2 has spectral radius of the order c^2. As a result, the spectral radius of the matrix (4.15) grows like c. Therefore, for the scheme Prolate+PC3, the time-step h decreases linearly with a. On the other hand, for the scheme Chebyshev+PC3, the time-step h decreases roughly as a^2, due to the $O(m^4)$ growth of the spectral radius of D.

2. Compared to the schemes Chebyshev+PC3 and Chebyshev+RK4, the scheme Prolate+PC3 demonstrates speed-up across all values of a. In addition, the speed-up increases with a because of the increasingly strict stability requirement imposed by the Chebyshev differentiation matrices as the wavenumber of the problem increases. For $a = 4$, the CPU time T it takes for the scheme Prolate+PC3 to achieve an accuracy of 10^{-12} is about 2.4 times less than it takes for the scheme Chebyshev+PC3; while for $a = 399$, the speed-up becomes about 130 times.

3. It should be noted, however, that in the computing environment in which the experiments are run (as described at the beginning of Section 4.2), the CPU time it takes for the direct application of an $N \times N$ differentiation matrix does not scale quadratically across all values of N. In particular, the proportionality constant for the direct application of matrices of size $N > 600$ is somewhat larger than that associated with $N < 600$. This phenomenon is related to the memory management

of the particular computing environment. Consequently, for $a = 199$, the scheme Prolate+PC3, with differentiation matrix D of size $m = 445$, demonstrates a disproportionate amount of speed-up (about 300 times) when compared to the scheme Chebyshev+PC3, with D of size $N = 698$. Therefore, the performance comparison should be seen more as an indication than as an absolute measure of their relative performance.

a	t_f	c	ε	m	ρ	n_s	h	E_{l_2}	T_c	T
4	55.278	14.5	10^{-14}	31	1.76E+04	22 210	2.49E-03	2.63E-12	0.01	0.26
9	26.184	30.0	10^{-13}	42	4.84E+04	17 500	1.50E-03	2.63E-12	0.02	0.30
19	12.756	61.5	10^{-14}	68	2.16E+05	18 000	7.08E-04	2.03E-12	0.05	0.60
49	5.025	156	10^{-13}	133	1.08E+06	15 900	3.16E-04	1.54E-12	0.26	1.61
99	2.500	315	10^{-13}	238	3.86E+06	15 000	1.67E-04	1.73E-12	1.26	4.31
199	1.247	628	10^{-14}	445	1.75E+07	15 900	7.84E-05	1.78E-12	6.52	15.28
399	0.623	1260	10^{-13}	847	6.06E+07	14 800	4.21E-05	8.40E-13	42.42	253.94

Table 15: Performance of the scheme Prolate+PC3 in the solution of (4.16)-(4.18).

Table 16: Performance of the scheme Chebyshev+PC3 in the solution of (4.16)-(4.18).

a	t_f	m	ρ	n_s	h	E_{l_2}	T
4	55.278	37	7.99E+04	47 500	1.16E-03	1.49E-12	0.63
9	26.184	58	5.01E+05	56 200	4.66E-04	1.67E-12	1.44
19	12.756	95	3.70E+06	74 400	1.71E-04	1.71E-12	4.04
49	5.025	206	8.37E+07	131 500	3.82E-05	1.35E-12	26.69
99	2.500	368	8.59E+08	222 100	1.13E-05	1.27E-12	156.92
199	1.247	700	1.13E+10	402 000	3.10E-06	7.42E-13	4526.4
399	0.623	1350	1.57E+11	747 500	8.33E-07	1.35E-12	33 003

Table 17: Performance of the scheme Chebyshev+RK4 in the solution of (4.16)-(4.18).

a	t_f	m	ρ	n_s	h	E_{l_2}	T
4	55.278	36	7.14E+04	600 000	9.21E-05	4.04E-12	3.03
9	26.184	58	5.01E+05	800 000	3.27E-05	4.16E-12	9.78
19	12.756	96	3.86E+06	800 000	1.60E-05	4.94E-12	26.83
49	5.025	206	8.37E+07	800 000	6.28E-06	5.85E-12	157.68
99	2.500	368	8.59E+08	800 000	3.13E-06	6.15E-12	549.76
199	1.247	700	1.13E+10	800 000	1.56E-06	5.81E-12	10 953
399	0.623	1350	1.57E+11	800 000	7.79E-07	6.28E-12	45 637

4.2.2 Eigenvalues of the Bessel's equation

In this subsection, we consider the eigenvalues of the Bessel's equation:

$$u_{rr} + \frac{1}{r}u_r - \frac{l^2}{r^2}u = \lambda u, \quad r \in [0,1], \quad l = 0, 1, 2, \ldots, \tag{4.20}$$

with the boundary conditions

$$u'(0) = 0, u(1) = 0 \quad \text{if } l = 0, \tag{4.21}$$

$$u(0) = 0, u(1) = 0 \quad \text{if } l \neq 0. \tag{4.22}$$

The equation (4.20) arises from the solution of the wave equation in polar coordinates. For each $l \geq 0$, the exact eigenvalues $\lambda_{l,k},\ k = 1, 2, \ldots$ of (4.20) are the negative of the squares of the roots of the lth order Bessel function J_l; in other words,

$$\lambda_{l,k} = -j_{l,k}^2, \tag{4.23}$$

where

$$J_l(j_{l,k}) = 0, \quad k = 1, 2, \ldots. \tag{4.24}$$

The corresponding eigenfunctions are

$$u_k(r) = J_l(j_{l,k}r), \quad k = 1, 2, \ldots. \tag{4.25}$$

In the following, we compute numerically the eigenvalues of the Bessel's equation (4.20) using the scheme described in Section 3.2, the Chebyshev collocation method, and the fourth-order finite difference scheme (FD4), and compare them to the exact eigenvalues (4.23). More precisely, we compute the eigenvalues of the $m \times m$ matrix

$$M = D^{(2)} + AD^{(1)} - B, \tag{4.26}$$

where $D^{(1)}$ and $D^{(2)}$ are the first and second derivative matrices incorporating the boundary conditions (4.21) or (4.22), and A and B are diagonal matrices with diagonal entries

$$A_{i,i} = 1/x_i, \tag{4.27}$$

$$B_{i,i} = l^2/x_i, \quad i = 1, 2, \ldots, m \tag{4.28}$$

where $x_1, \ldots, x_m$ are the nodes used in the construction of $D^{(1)}$ and $D^{(2)}$.

Remark 4.4. For the scheme described in Section 3.2, the boundary conditions (4.21) for the case $l = 0$ is incorporated into the orthonormal set of functions $\{\phi_i\}$ (3.18) by subtracting from the PSWFs $\psi_0, \ldots, \psi_{n-1}$ the linear functions

$$\mu_j(x) = \psi_j(1) + \psi_j'(-1)(x - 1), \quad j = 0, \ldots, n - 1, \tag{4.29}$$

and then applying the Gram-Schmidt algorithm to the matrix (3.17).

Remark 4.5. For the Chebyshev collocation method, we incorporate the boundary conditions (4.21) into $D^{(1)}$ and $D^{(2)}$ of (4.26) explicitly using the last row and column of the $m \times m$ Chebyshev matrix $D := \hat{D}_C^{(1)}$ given by formulas (2.47) and (2.50). In particular, $D^{(1)}$ is constructed by the formula

$$D^{(1)} = D_S + uv^T, \tag{4.30}$$

where D_S is the $(m-2) \times (m-2)$ matrix obtained by removing from D its first and last rows and columns, and u and v are vectors with entries:

$$u_i = D_{i+1,m}, \tag{4.31}$$

$$v_i = -D_{m,i+1}/D_{m,m}, \tag{4.32}$$

for $i = 1, \ldots, m-2$. A similar procedure is used to incorporate (4.21) into the differentiation matrices of the FD4 scheme.

First, we look at the eigenvalues of (4.20) for the case $l = 0$. The graph of the Bessel function J_0 on the interval $[0, 160]$ is shown in Figure 16. Table 18 shows the non-zero eigenvalues of the matrix M in (4.26) constructed using the scheme described in Section 3.2 with $c = 16\pi$ and $\varepsilon = 10^{-13}$. There are 56 non-zero eigenvalues, which is equal to the number of functions in the orthonormal set $\{\phi_i\}$ (3.18) corresponding to the chosen c and ε. The eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{56}$ are real, distinct, and negative; and $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{32}$ are accurate to 13 digits with respect to the exact eigenvalues (4.23). This is expected, given the choice $c = 16\pi$ and the form of the exact eigenfunctions:

$$u_k(r) = J_0(j_{0,k} r), \tag{4.33}$$

which contains $k/2$ wavelengths on the interval $[0, 1]$, for $k = 1, 2, \ldots$. For $k > 32$, the magnitudes of $\tilde{\lambda}_k$ starts to grow exponentially, and they no longer approximate (4.23).

For fixed ε, the spectral radius ρ of the matrix M grows as c^2. As an illustration, Figure 17 plots the spectral radii of M against c, for $\varepsilon = 10^{-7}$ and 10^{-13}. All eigenvalues of M computed are either zero, or real and negative. On the other hand, the eigenvalues of M constructed by the Chebyshev collocation method are all real and negative. The spectral radii grow as m^4, which is illustrated in Figure 18.

Remark 4.6. For the scheme described in Section 3.2, since the prolate quadrature nodes constructed with the algorithm of [69] do not include the endpoints of the interval (see Remark 3.1), there is no division by zero in the construction of the matrices A and B in (4.26). On the other hand, for the Chebyshev collocation and the FD4 schemes, we explicitly omit the endpoints in the construction of $D^{(1)}$ and $D^{(2)}$ (see Remark 4.5), so division by zero in (4.26) is avoided.

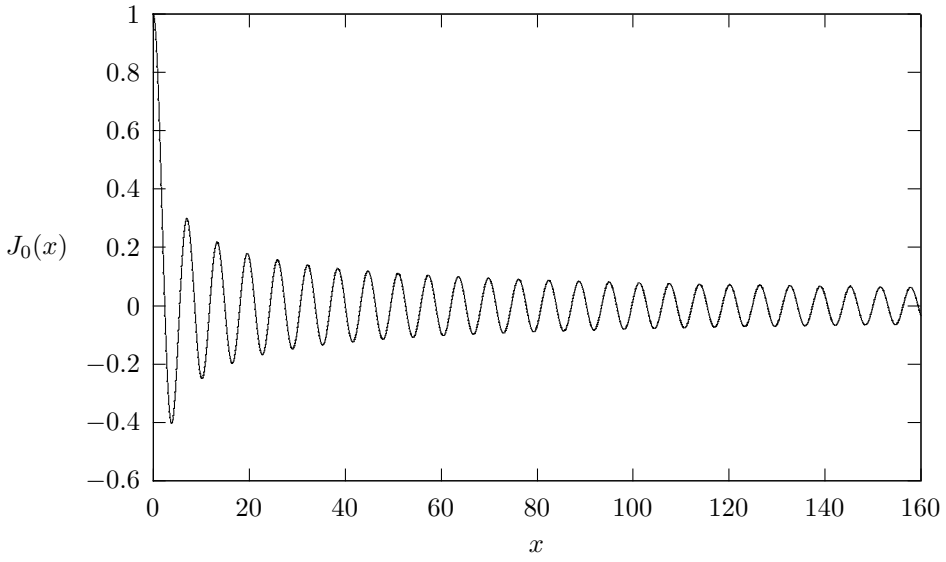

Figure 16: Plot of the Bessel function $J_0(x)$.

Table 18: Non-zero eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{56}$ of the matrix M constructed by the scheme described in Section 3.2 using $c = 16\pi$ and $\varepsilon = 10^{-13}$. $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{32}$ are accurate to 13 digits with respect to (4.23).

-.5783185962945060E+01	-.3047126234366095E+02
-.7488700679069376E+02	-.1390402844264589E+03
-.2229323036176315E+03	-.3265633529323268E+03
-.4499335285180342E+03	-.5930428696559549E+03
-.7558913947839302E+03	-.9384791134756928E+03
-.1140806031099643E+04	-.1362872150854103E+04
-.1604677474740231E+04	-.1866222004061850E+04
-.2147505739697837E+04	-.2448528682258052E+04
-.2769290832176346E+04	-.3109792189768252E+04
-.3470032755267547E+04	-.3850012528850579E+04
-.4249731510652221E+04	-.4669189700777149E+04
-.5108387099307645E+04	-.5567323706308965E+04
-.6045999521833543E+04	-.6544414545923878E+04
-.7062568778614452E+04	-.7600462219933994E+04
-.8158094869905980E+04	-.8735466728550498E+04
-.9332577795883790E+04	-.9949428071920061E+04
-.1058601755667020E+05	-.1124234625029806E+05
-.1191841415922765E+05	-.1261422120220677E+05
-.1332976659249901E+05	-.1406505654372051E+05
-.1482021199418147E+05	-.1559529716614024E+05
-.1639970878090875E+05	-.1723462950822729E+05
-.1824687473444598E+05	-.1928317458550053E+05
-.2100242484019345E+05	-.2250212957245788E+05
-.2578938634166968E+05	-.2810174282886851E+05
-.3457735371967172E+05	-.3863584374915628E+05
-.5387757175873770E+05	-.6174489536835697E+05
-.1040345004919501E+06	-.1276690375260464E+06
-.4272322530955649E+06	-.4908888139141377E+06

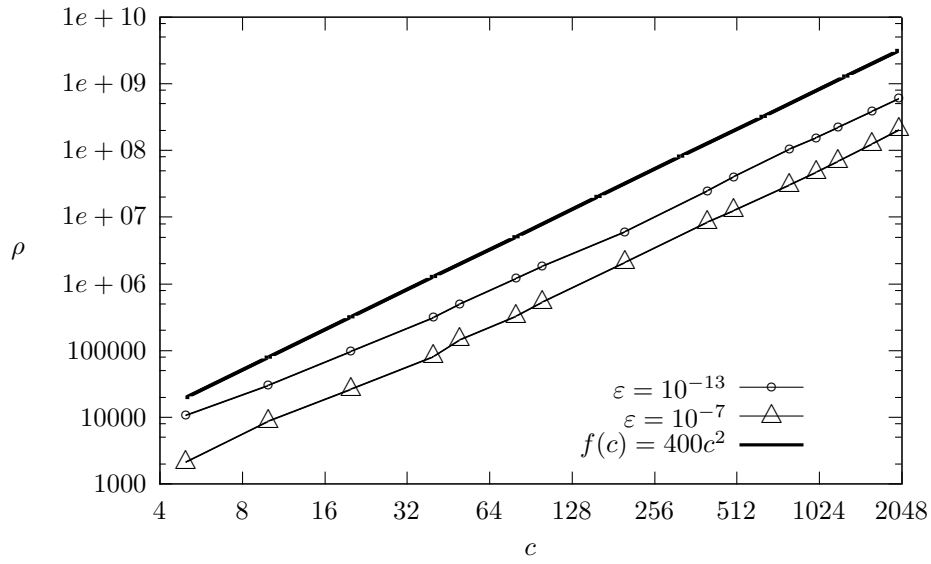

Figure 17: Log-log plot of the spectral radius ρ of the matrix M constructed by the scheme in Section 3.2 against c, with $\varepsilon = 10^{-7}, 10^{-13}$.

Next, we look at the convergence performance of the scheme described in Section 3.2, the Chebyshev collocation scheme, and the FD4 scheme in approximating the exact eigenvalues (4.23) of the Bessel's equation. The column labeled 'PSWFs' in Table 19 shows the relative errors E_{rel} when the eigenvalue $\lambda_{0,40}$ is computed using the scheme described in Section 3.2 with $\varepsilon = 10^{-13}$ and varying values of c. For comparison, Table 19 also shows the errors E_{rel} when $\lambda_{0,40}$ is computed using the Chebyshev collocation and the FD4 schemes. Also, Figure 19 shows the plot of the errors E_{rel} in Table 19 against the number of nodes m. In addition, Figures 20-22 show the plots of the errors E_{rel} against m when the eigenvalues $\lambda_{1,10}$, $\lambda_{10,100}$, and $\lambda_{50,200}$ are computed using the above three schemes. From Figures 19-22, we see that both the scheme described in Section 3.2 and the Chebyshev collocation scheme display exponential (spectral) convergence in accuracy. However, considering the frequency of the corresponding eigenfunctions, the scheme described in Section 3.2 requires fewer points per wavelength to achieve double precision accuracy. For instance, for the computation of $\lambda_{50,200}$, it takes about 2.6 points per wavelength for the scheme described in Section 3.2 to achieve 15 digits of accuracy, while the Chebyshev collocation scheme takes about 4 points per wavelength to achieve the same order of accuracy. The FD4 scheme, on the other hand, only displays 1 to 2 digits of accuracies with comparable numbers of points.

Remark 4.7. We would like to reiterate that for the scheme described in Section 3.2, the number of nodes m is determined by the choice of c and ε. We plot in Figures 19-22 the relative errors E_{rel} obtained by the scheme against m for ease of comparison to the Chebyshev collocation and the FD4 schemes.

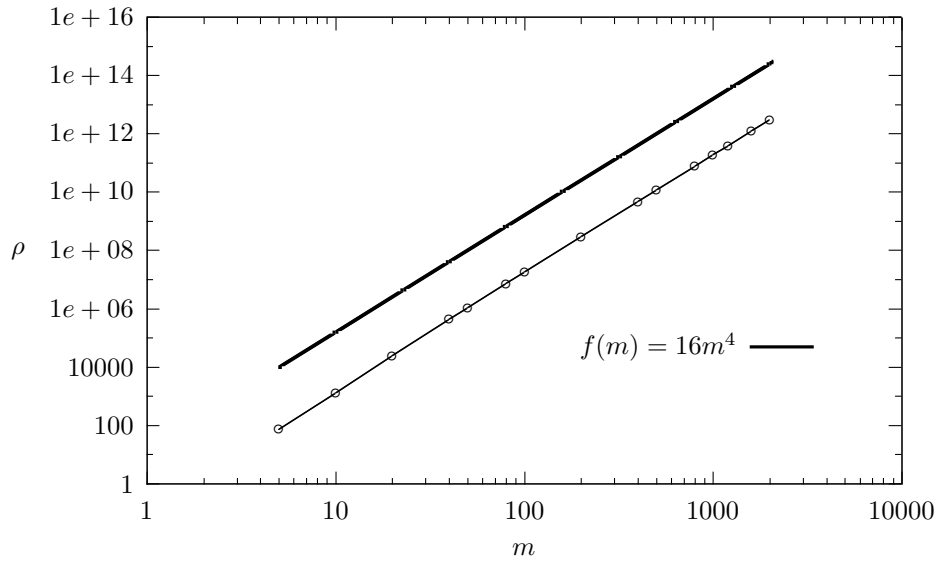

Figure 18: Log-log plot of the spectral radius ρ of the matrix M constructed by the Chebyshev collocation method against the number of nodes m.

Table 19: Relative errors E_{rel} of computing $\lambda_{0,40}$ using the scheme described in Section 3.2, the Chebyshev collocation scheme, and the FD4 scheme.

	PSWFs		Chebyshev		FD4	
c	m	E_{rel}	m	E_{rel}	m	E_{rel}
48	56	1.01E-03	68	1.34E-03	56	2.02E-01
50	59	3.60E-05	72	3.39E-05	63	1.51E-01
52	60	8.38E-07	76	4.69E-07	70	1.17E-01
54	62	1.94E-07	80	9.92E-09	77	7.66E-02
56	63	9.92E-08	84	3.46E-10	84	4.64E-02
58	65	2.57E-10	88	1.05E-11	91	3.29E-02
60	66	1.52E-11	92	2.54E-13	98	2.42E-02
62	68	5.72E-15	96	3.15E-15	105	1.82E-02

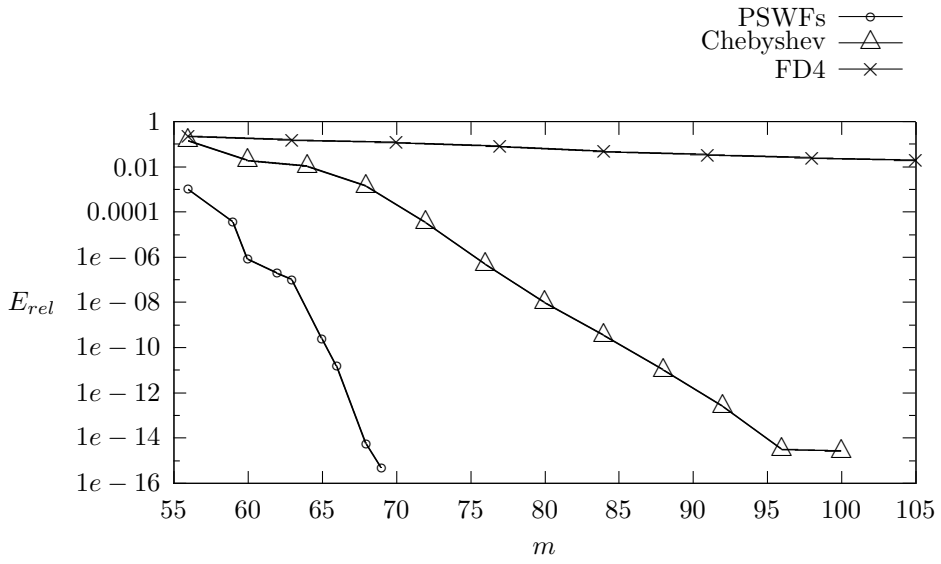

Figure 19: Relative errors E_{rel} against m in the computation of $\lambda_{0,40}$.

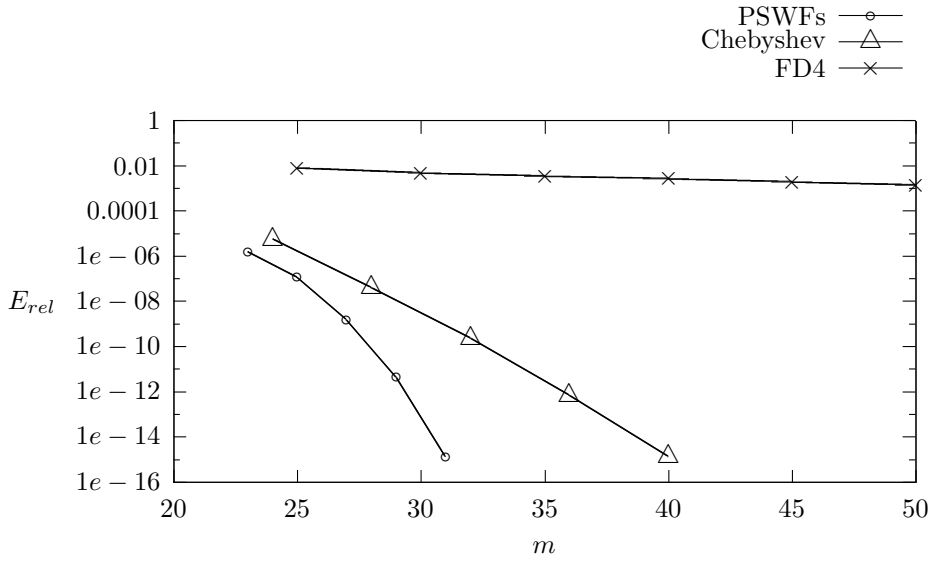

Figure 20: Relative errors E_{rel} against m in the computation of $\lambda_{1,10}$.

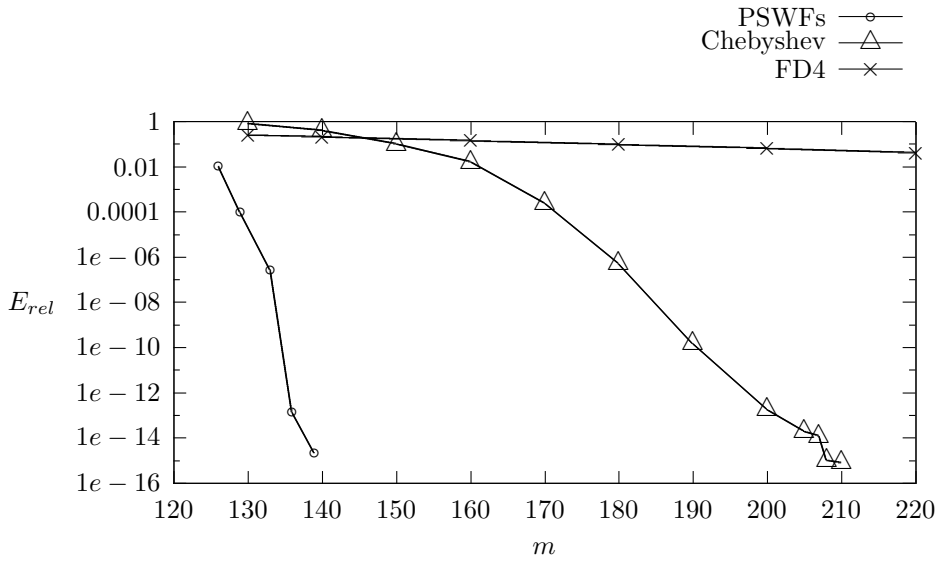

Figure 21: Relative errors E_{rel} against m in the computation of $\lambda_{10,100}$.

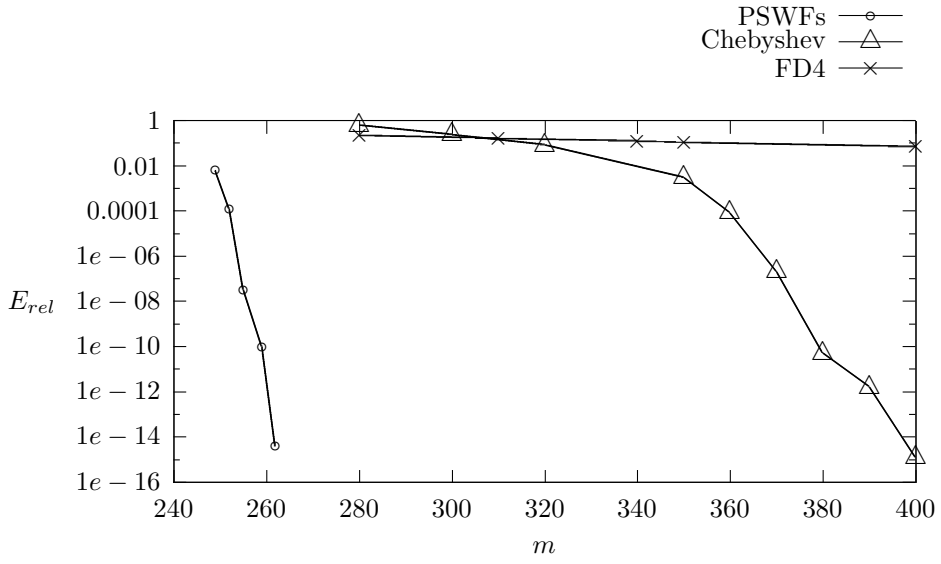

Figure 22: Relative errors E_{rel} against m in the computation of $\lambda_{50,200}$.

4.2.3 A variable-coefficient hyperbolic equation

In this subsection, we consider the following variable-coefficient hyperbolic equation:

$$u_t = a(x)u_x, \tag{4.34}$$

$$u(x,0) = \sin(30\pi x + 1.5\sin(8.9\pi x)), \tag{4.35}$$

$$u(1,t) = g(t), \tag{4.36}$$

where $x \in [-1,1]$, $t \geq 0$, and

$$a(x) = \frac{1}{30\pi + 13.35\pi\cos(8.9\pi x)}, \tag{4.37}$$

$$g(t) = \sin(30\pi + 1.5\sin(8.9\pi) + t). \tag{4.38}$$

The analytical solution of (4.34)-(4.38) is given by

$$u(x,t) = \sin(30\pi x + 1.5\sin(8.9\pi x) + t). \tag{4.39}$$

Using the transformation

$$v(x,t) = u(x,t) - g(t), \tag{4.40}$$

we reduce (4.34)-(4.36) into a hyperbolic equation with zero boundary condition at $x = 1$:

$$v_t = a(x)v_x - g'(t), \tag{4.41}$$

$$v(x,0) = \sin(30\pi x + 1.5\sin(8.9\pi x)) - g(0), \tag{4.42}$$

$$v(1,t) = 0. \tag{4.43}$$

In the following, we solve (4.41)-(4.43) numerically on the time interval [0, 1000] using the schemes Prolate+PC1, Prolate+PC3, Chebyshev+PC1, Chebyshev+PC3, and Chebyshev+RK4, and compare their timings and accuracies. For the schemes Prolate+PC1 and Prolate+PC3, we construct the matrix M discretizing the operator $a \cdot v_x$ by first constructing an $m \times m$ first derivative matrix D incorporating the boundary condition $u(1) = 0$ using the scheme described in Section 3.2, and then multiplying D on the left by the $m \times m$ diagonal matrix A with diagonal entries

$$A_{i,i} = a(x_i),\ i = 1, \ldots, m, \tag{4.44}$$

where $x_1, \ldots, x_m$ are the nodes used in the construction of D. The matrices M are similarly constructed for the schemes Chebyshev+PC1, Chebyshev+PC3, and Chebyshev+RK4, in which we construct the first derivative matrix $\hat{D}_C$ using the Chebyshev collocation method described in Section 2.3.2, and then strip $\hat{D}_C$ of its first row and column.

Tables 20-24 summarize the results for the schemes Prolate+PC1, Prolate+PC3, Chebyshev+PC1, Chebyshev+PC3, and Chebyshev+RK4 respectively. In the tables, m denotes the number of nodes

on $[-1,1]$ used in the construction of the matrix M, ρ denotes the spectral radius of the discretized operator $a \cdot v_x$, and E_{l_2} denotes the relative l_2 error of the numerical solution of v at the final time $t_f = 1000$. The rest of the notation is the same as that in Tables 9-13 in Section 4.2.1. Figure 23 shows, for the above schemes, the CPU time T versus the relative error E_{l_2}, while Figure 24 shows the number of time steps n_s versus E_{l_2}. All data points in Figures 23 and 24 are selected from the results in Tables 20-24. In addition, Figures 25 and 26 show, for selected sets of parameters, the spectra of the matrix M constructed using the scheme described in Section 3.2 and the Chebyshev collocation method, respectively.

In the following, we make several observations and comments based on the presented results, and on the results of our more extensive experiments.

1. In all of the above schemes, the eigenvalues of the matrix M lie on the left half-plane, so stability is guaranteed provided that sufficiently small time-steps are chosen for the time-marching schemes PC1, PC3, and RK4. In particular, for the schemes Prolate+PC1, Prolate+PC3, Chebyshev+PC1, and Chebyshev+PC3, the number of steps n_s is always dominated by the stability requirement. On the other hand, for the scheme Chebyshev+RK4, n_s is dominated by the stability requirement up to a desired accuracy of about 10^{-8}; after that, n_s is dominated by the accuracy requirement.

2. Compared to the schemes Chebyshev+PC1, Chebyshev+PC3, and Chebyshev +RK4, the schemes Prolate+PC1 and Prolate+PC3 are superior across all accuracies, both in terms of CPU time T and the number of time steps n_s. This is because, as indicated in Tables 20-24, the matrices M constructed by the scheme described in Section 3.2 have smaller spectral radii ρ compared to those constructed using the Chebyshev collocation method, given the same accuracy requirement. In particular, our more extensive experiments show that the matrices M constructed by the former scheme have ρ that grow as c for fixed ε, while those constructed by the latter scheme have ρ that grow as m^2.

3. From Table 21, we see that the bandlimit parameter c required to solve for v in (4.40) to precision 10^{-13} is about 480. This corresponds to about 153 wavelengths on the interval $[-1,1]$. Thus, the frequency of v in the spatial dimension is much higher than its frequency in the time dimension, the latter of which equals $1/2\pi$. As a result, the penalty imposed by the stability requirement in this example is higher than that in the example of Section 4.2.1. In particular, the solution v spans about 160 periods on the time interval $[0,1000]$, which means that it takes about 130 and 590 steps per period for the schemes Prolate+PC1 and Prolate+PC3 to achieve an accuracy of 5×10^{-7} and 10^{-13}, respectively. These results, although less than optimal, still compare favorably to those obtained by the schemes Chebyshev+PC1, Chebyshev+PC3, and Chebyshev+RK4.

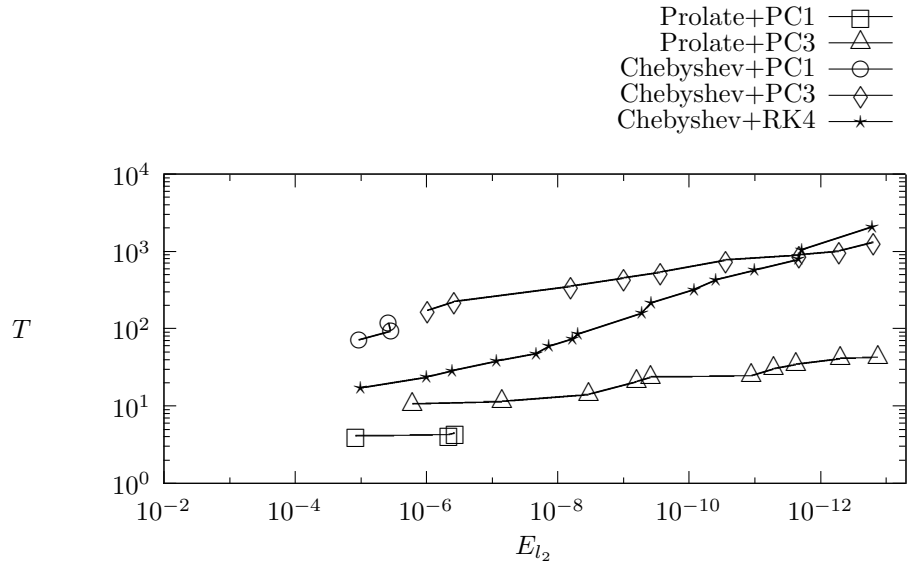

Figure 23: CPU time T versus relative error E_{l_2} in the solution of (4.41)-(4.43).

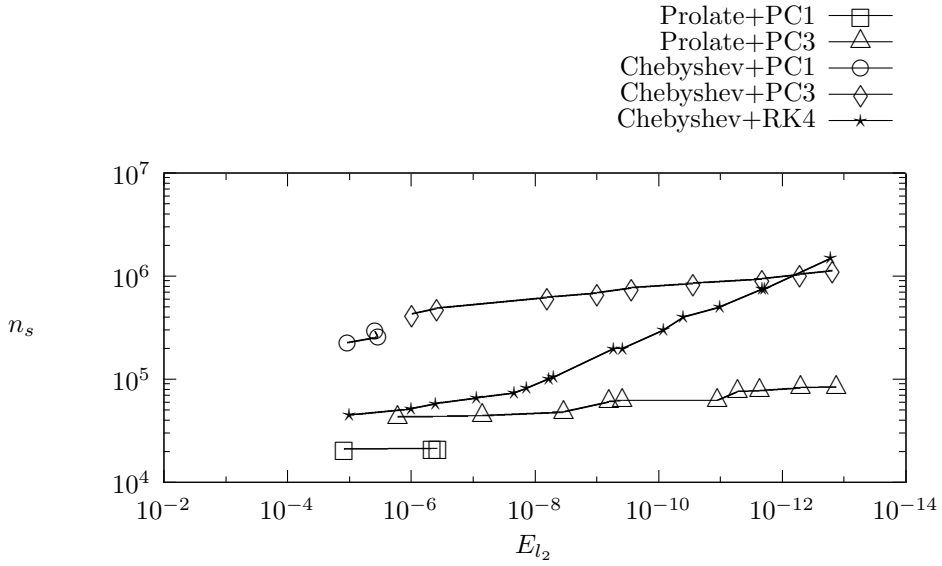

Figure 24: Number of time steps n_s versus relative error E_{l_2} in the solution of (4.41)-(4.43).

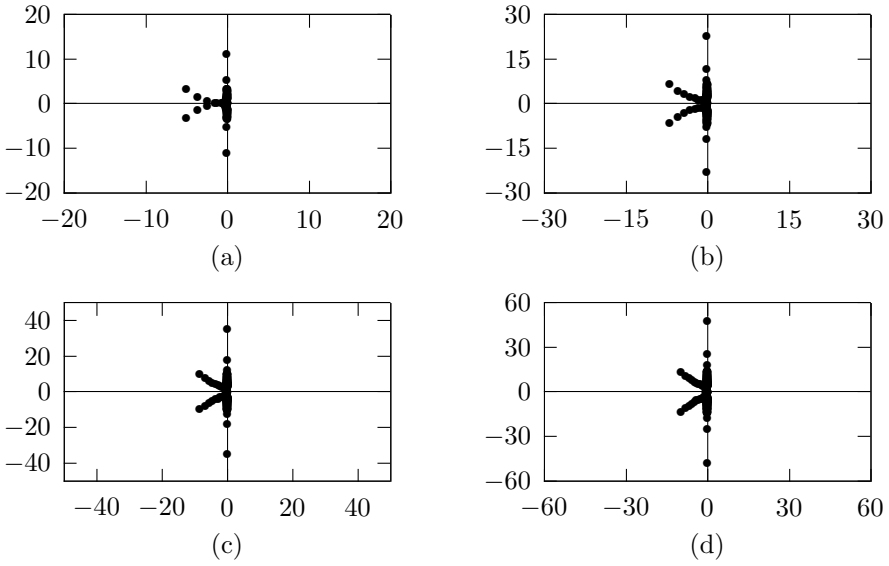

Figure 25: Spectra of M constructed using the scheme described in Section 3.2, with $\varepsilon = 10^{-13}$ and (from (a) to (d)) $c = 200, 400, 600, 800$.

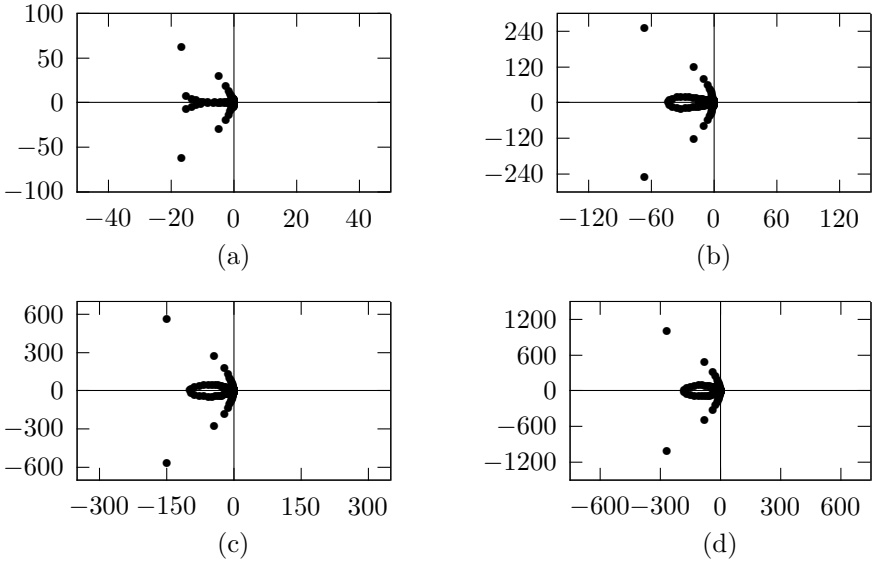

Figure 26: Spectra of M constructed using the Chebyshev collocation scheme, with (from (a) to (d)) $m = 200, 400, 600, 800$.

Table 20: Performance of the scheme Prolate+PC1 in the solution of (4.41)-(4.43).

c	ε	m	ρ	n_s	h	E_{l_2}	T_c	T
300	10^{-8}	217	1.15E+01	21 000	4.76E-02	1.20E-05	0.79	4.12
310	10^{-8}	226	1.16E+01	21 200	4.72E-02	4.61E-07	0.86	4.24
320	10^{-8}	230	1.18E+01	21 400	4.67E-02	3.69E-07	0.92	4.52
330	10^{-8}	237	1.19E+01	21 600	4.63E-02	5.44E-07	1.01	4.86

Table 21: Performance of the scheme Prolate+PC3 in the solution of (4.41)-(4.43).

c	ε	m	ρ	n_s	h	E_{l_2}	T_c	T
330	10^{-9}	242	1.41E+01	43 000	2.33E-02	1.61E-06	1.02	10.56
350	10^{-9}	252	1.44E+01	44 000	2.27E-02	7.00E-08	1.12	11.53
370	10^{-9}	268	1.58E+01	48 000	2.08E-02	3.36E-09	1.32	14.24
390	10^{-11}	283	2.00E+01	61 000	1.64E-02	6.28E-10	1.51	20.63
410	10^{-11}	299	2.04E+01	62 000	1.61E-02	3.79E-10	1.70	23.64
420	10^{-11}	305	2.05E+01	62 500	1.60E-02	1.11E-11	1.76	24.57
430	10^{-13}	314	2.51E+01	76 000	1.32E-02	5.15E-12	1.90	30.50
450	10^{-13}	330	2.55E+01	77 500	1.29E-02	2.31E-12	2.09	34.55
470	10^{-13}	343	2.74E+01	83 500	1.20E-02	4.99E-13	2.31	41.26
480	10^{-13}	349	2.76E+01	84 000	1.19E-02	1.30E-13	2.42	42.56

Table 22: Performance of the scheme Chebyshev+PC1 in the solution of (4.41)-(4.43).

m	ρ	n_s	h	E_{l_2}	T
280	1.27E+02	226 000	4.42E-03	1.06E-05	72.21
300	1.45E+02	256 000	3.91E-03	3.43E-06	93.32
320	1.66E+02	291 500	3.43E-03	3.81E-06	120.92
340	1.87E+02	330 000	3.03E-03	4.16E-06	154.23

Table 23: Performance of the scheme Chebyshev+PC3 in the solution of (4.41)-(4.43).

m	ρ	n_s	h	E_{l_2}	T
300	1.45E+02	430 000	2.33E-03	9.69E-07	173.22
320	1.66E+02	490 000	2.04E-03	3.77E-07	226.92
340	1.87E+02	560 000	1.79E-03	7.28E-07	289.54
360	2.10E+02	620 000	1.61E-03	6.32E-09	358.53
380	2.34E+02	690 000	1.45E-03	9.84E-10	435.70
400	2.59E+02	770 000	1.30E-03	2.73E-10	542.54
420	2.86E+02	855 000	1.17E-03	2.77E-11	776.64
440	3.14E+02	940 000	1.06E-03	2.09E-12	911.67
460	3.43E+02	1 040 000	9.62E-04	5.11E-13	1014.6
480	3.73E+02	1 130 000	8.85E-04	1.52E-13	1328.2

Table 24: Performance of the scheme Chebyshev+RK4 in the solution of (4.41)-(4.43).

m	ρ	n_s	h	E_{l_2}	T
280	1.27E+02	45 000	2.22E-02	1.00E-05	17.25
300	1.45E+02	52 000	1.92E-02	9.78E-07	23.81
320	1.67E+02	58 000	1.72E-02	3.99E-07	28.92
340	1.87E+02	66 000	1.52E-02	8.56E-08	37.97
360	2.10E+02	74 000	1.35E-02	2.11E-08	48.39
380	2.34E+02	82 000	1.22E-02	1.36E-08	59.50
		102 000	9.80E-03	5.89E-09	75.02
400	2.59E+02	105 000	9.52E-03	4.96E-09	84.34
		200 000	5.00E-03	5.21E-10	159.42
420	2.86E+02	200 000	5.00E-03	3.74E-10	214.88
		300 000	3.33E-03	8.19E-11	321.41
		400 000	2.50E-03	3.87E-11	429.48
440	3.14E+02	500 000	2.00E-03	1.01E-11	574.66
		750 000	1.33E-03	3.11E-12	867.72
460	3.43E+02	750 000	1.33E-03	2.11E-12	788.76
		1 000 000	1.00E-03	9.33E-13	1054.3
480	3.74E+02	750 000	1.33E-03	1.90E-12	1055.8
		1 500 000	6.67E-04	1.62E-13	2086.1

5 Conclusions and future work

We have presented a new class of numerical differentiation schemes constructed via the PSWFs. As opposed to existing collocation methods, the schemes are based on the construction of an approximate interpolation u of a function f via a least-squares type procedure, in which we do not require u to be exactly equal to f at any of the interpolation nodes. For problems that involve bandlimited functions, the schemes require fewer points per wavelength to attain the same accuracy when compared to the Chebyshev collocation method. In addition, when solving time-dependent PDEs with non-periodic boundary conditions, the resulting first and second derivatives have spectral radii that grow as c and c^2 respectively, for fixed precision ε. Our numerical experiments indicate that, when combined with a numerical ODE solver to solve time-dependent PDEs, the schemes outperform the Chebyshev collocation and the finite difference methods, in particular when high accuracy is required or the solutions contain large numbers of wavelengths.

In the following, we discuss several possible extensions to the schemes:

1. It is possible to accelerate the application of the differentiation matrices constructed by the schemes to vectors. In particular, the differentiation matrices D of the schemes presented here satisfy the symmetric property

   $$D_{i,j} = -D_{m-i,m-j},\ 1 \leq i,j \leq m,$$

   in the case of first derivatives, and

   $$D_{i,j} = D_{m-i,m-j},\ 1 \leq i,j \leq m,$$

   in the case of second derivatives, where m is the dimension of D. Thus, a fast algorithm for applying D to a vector v that is similar to the scheme of [60] can be constructed.

2. On the other hand, the schemes presented here can be modified to make the resulting differentiation matrices amenable to fast applications. For pseudospectral methods based on Chebyshev or Legendre polynomials, the fast Fourier transform (FFT) or the fast multipole method (FMM) (see, for instance, [12, 31, 33]) can be employed to reduce the cost of applying an $N \times N$ differentiation matrix to a vector to $O(N \log N)$ operations (see [8, 20, 63]). For pseudospectral methods based on the PSWFs, the authors of [39] constructed an algorithm that utilizes the fast multipole method (FMM) to reduce the cost to $O(N)$ operations. Similar modifications to our schemes are currently under investigation by the authors, and any results will be reported in a later date.

3. For the schemes presented here, the interpolation nodes on which the differentiation matrix D is constructed all lie in the interior of the interval $[-1, 1]$ (see Remark 3.1), and boundary conditions are incorporated implicitly in the orthonormal set of functions $\phi_1, \ldots, \phi_k$ (see (3.14)). This may seem to pose difficulty in some two-dimensional problems, such as the two-dimensional wave equation

   $$u_{tt} = u_{xx} + u_{yy},\ \ -1 \leq x,y \leq 1, \tag{5.1}$$

with boundary conditions

$$u(x,y) = \begin{cases} \sin(\pi x) & \text{for } y = \pm 1, \\ 0 & \text{for } x = \pm 1, \end{cases} \tag{5.2}$$

since then the boundary conditions cannot be incorporated independently into the differentiation matrices discretizing u_{xx} and u_{yy}.

One possible way to tackle it is to modify the differentiation matrix D in (3.22):

$$D = UP^*W \tag{5.3}$$

into the form

$$\tilde{D} = YDX, \tag{5.4}$$

where X is a matrix that interpolates from a set of points $y_1, \ldots, y_l$ to the quadrature nodes $x_1, \ldots, x_m$ used in the construction of D, with $y_1, \ldots, y_l$ containing the end points of the interval $[-1, 1]$; and Y is a matrix that interpolates from $x_1, \ldots, x_m$ to $y_1, \ldots, y_l$. Both X and Y can be constructed via least-squares. By using the matrix $\tilde{D}$ instead of D in a time-marching scheme, the boundary conditions can be enforced directly at each time-step.

In addition, using $\tilde{D}$ as the differentiation matrix allows the incorporation of boundary conditions via the method described in Remark 4.5, which can be more convenient in certain problems.

References

[1] W. F. Ames, *Numerical methods for partial differential equations*, Academic Press, San Diego, 1992.

[2] R. Baltensperger, J. P. Berrut, *The errors in calculating the pseudospectral differentiation matrices for Chebyshev-Gauss-Lobatto points*, Comput. Math. Appl. 37 (1999) 41-48.

[3] A. Bayliss, A. Class, B. J. Matkowsky, *Roundoff error in computing derivatives using the Chebyshev differentiation matrix*, J. Comput. Phys. 116 (1994) 380-383.

[4] A. Bayliss, E. Turkel, *Mappings and accuracy for Chebyshev pseudospectral approximations*, J. Comput. Phys. 101 (1992) 349-359.

[5] G. Beylkin, L. Monzón, *On generalized Gaussian quadratures for exponentials and their applications*, Appl. Comput. Harmon. Anal. 12 (2002) 332-373.

[6] G. Beylkin, K. Sandberg, *Wave propagation using bases for bandlimited functions*, Wave Motion 41 (2005) 263-291.

[7] A. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[8] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, Dover, New York, 2001.

[9] J. P. Boyd, *Prolate spheroidal wavefunctions as an alternative to Chebyshev and Legendre polynomials for spectral element and pseudospectral algorithms*, J. Comput. Phys. 199 (2004) 688-716.

[10] K. S. Breuer, R. M. Everson, *On the errors incurred calculating derivatives using Chebyshev polynomials*, J. Comput. Phys. 99 (1992) 56-67.

[11] C. Canuto, M. Y. Hussaini, A. Quarteroni, T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, Berlin, 1988.

[12] J. Carrier, L. Greengard, V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Stat. Comput. 9(4) (1988) 669-686.

[13] Q.-Y. Chen, D. Gottlieb, J. S. Hesthaven, *Spectral methods based on prolate spheroidal wave functions for hyperbolic PDEs*, SIAM J. Numer. Anal 43 (5) (2005) 1912-1933.

[14] H. Cheng, N. Yarvin, V. Rokhlin, *Non-linear optimization, quadrature, and interpolation*, SIAM J. Optim. 9 (4) (1999) 901-923.

[15] G. Dahlquist, A. Bjorck, *Numerical Methods*, Dover, New York, 1974.

[16] G. D. de Villiers, F. B. T. Marchaud, E. R. Pike, *Generalized Gaussian quadrature applied to an inverse problem in antenna theory*, Inverse Problems, 17 (4) (2001) 1163-1179.

[17] W. S. Don, A. Solomonoff, *Accuracy and speed in computing the Chebyshev collocation derivative*, SIAM J. Sci. Comput. 16 (6) (1995) 1253-1268.

[18] W. S. Don, A. Solomonoff, *Accuracy enhancement for higher derivatives using Chebyshev collocation and a mapping technique*, SIAM J. Sci. Comput. 18 (4) (1997) 1040-1055.

[19] M. Dubiner, *Asymptotic analysis of spectral methods*, J. Sci. Comput. 2 (1) (1987) 3-31.

[20] A. Dutt, M. Gu, V. Rokhlin, *Fast algorithms for polynomial interpolation, integration, and differentiation*, SIAM J. Numer. Anal. 33 (1996) 1689-1711.

[21] S. J. Farlow, *Partial Differential Equations for Scientists and Engineers*, Dover, New York, 1993.

[22] J. H. Ferziger, *Numerical Methods for Engineering Application*, Wiley-Interscience, New York, 1998.

[23] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, 1996.

[24] D. Funaro, *Polynomial Approximation of Differential Equations*, Springer-Verlag, New York, 1992.

[25] F. Gantmacher, M. Krein, *Oscillation matrices and kernels and small oscillations of mechanical systems*, second edition, Moscow: Gosudarstv. Izdat. Tehn-Teor. Lit., 1950 (in Russian).

[26] A. Glaser, V. Rokhlin, *A new class of highly accurate solvers for ordinary differential equations*, J. Sci. Comput. 38 (2009) 368-399.

[27] D. Gottlieb, M. Y. Hussaini, S. A. Orszag, *Theory and applications of spectral methods*, Spectral Methods for Partial Differential Equations, R. G. Voigt, D. Gottlieb, and M. Y. Hussani, eds., Proc. 1982 NASA Langley Research Center Symposium, SIAM, Philadelphia, 1984, 1-54.

[28] D. Gottlieb, L. Lustman, *The Dufort-Frankel Chebyshev method for parabolic initial boundary value problems*, Computers and Fluids 11 (2) (1983) 107-120.

[29] D. Gottlieb, L. Lustman, *The spectrum of the Chebyshev collocation operator for the heat equation*, SIAM J. Numer. Anal. 20 (1983) 909-921.

[30] D. Gottlieb, S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, SIAM, Philadelphia, 1981.

[31] L. Greengard, V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys. 73 (1987) 325-348.

[32] J. S. Hesthaven, P. G. Dinsen, J. P. Lynov, *Spectral collocation time-domain modeling of diffractive optical elements*, J. Comput. Phys. 155 (1999) 287-306.

[33] T. Hrycak, V. Rokhlin, *An improved fast multipole algorithm for potential fields*, SIAM J. Sci. Comput. 19(6) (1998) 1804-1826.

[34] A. Iserles, *A First Course in the Numerical Analysis of Differential Equations*, Cambridge University Press, Cambridge, 1996.

[35] S. Karlin, *The existence of eigenvalues for integral operators*, Trans. Amer. Math. Soc. 113 (1964) 1-17.

[36] S. Karlin, W. Studden, *Tchebycheff Systems with Applications in Analysis and Statistics*, Wiley-Interscience, New York, 1966.

[37] D. Kosloff, H. Tal-ezer, *A modified Chebyshev pseudospectral method with an $O(N^{-1})$ time step restriction*, J. Comput. Phys. 104 (1993) 457-469.

[38] N. Kovvali, W. Lin, L. Carin, *Pseudospectral method based on prolate spheroidal wave functions for frequency-domain electromagnetic simulations*, IEEE Trans. Antennas Propag. 53 (12) (2005) 3990-4000.

[39] N. Kovvali, W. Lin, Z. Zhao, L. Couchman, L. Carin, *Rapid prolate pseudospectral differentiation and interpolation with the fast multipole method*, SIAM J. Sci. Comput. 28 (2) (2006) 485-497.

[40] M. G. Krein, *The ideas of P. L. Chebyshev and A. A. Markov in the theory of limiting values of integrals*, Amer. Math. Soc. Transl. 12 (1959) 1-122.

[41] H.-O. Kreiss, J. Lorenz, *Initial Boundary Value Problems and the Navier Stokes Equations*, Academic Press, San Diego, 1989.

[42] H. J. Landau, H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis, and uncertainty - II*, Bell Syst. Tech. J. 40 (1961) 65-84.

[43] H. J. Landau, H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis, and uncertainty - III: the dimension of space of essentially time- and band-limited signals*, Bell Syst. Tech. J. 41 (1962) 1295-1336.

[44] H. J. Landau, H. Widom, *Eigenvalue distribution of time and frequency limiting*, J. Math. Anal. Appl. 77 (1980) 469-481.

[45] W. Lin, N. Kovvali, L. Carin, *Pseudospectral method based on prolate spheroidal wave functions for semiconductor nanodevice simulation*, Comput. Phys. Commun. 175 (2006) 78-85.

[46] W. Lyons, H. D. Ceniceros, S. Chandrasekaran, M. Gu, *Fast algorithms for spectral collocation with non-periodic boundary conditions*, J. Comput. Phys. 207 (1) (2005) 173-191.

[47] J. Ma, V. Rokhlin, S. Wandzura, *Generalized Gaussian quadratures rules for systems of arbitrary functions*, SIAM J. Numer. Anal. 33 (1996) 971-996.

[48] A. Markov, *On the limiting value of integrals in connection with interpolation*, Zap. Imp. Akad. Nauk. Fix-Mat. Otd. 6 (1898) (in Russian).

[49] A. Markov, *Selected papers on continued fractions and the theory of functions deviating least from zero*, OGIZ, Moscow, Leningrad (1948) (in Russian).

[50] J. Mead, R. A. Renaut, *Accuracy, resolution and stability properties of a modified Chebyshev method*, SIAM J. Sci. Comput. 24 (1) (2002) 143-160.

[51] B. Mercier, *An Introduction to the Numerical Analysis of Spectral Methods*, Springer-Verlag, New York, 1989.

[52] P. M. Morse, J. Feshbach, *Methods of Theoretical Physics*, McGraw-Hill, New York, 1953.

[53] W. H. Press, S. A. Teulkolsky, S. A. Vetterling, W. T. Flannery, *Numerical Recipes in Fortran 77*, Cambridge University Press, Cambridge, 2001.

[54] W. E. Schiesser, *The Numerical Method of Lines: Integration of Partial Differential Equations*, Academic Press, San Diego, 1991.

[55] W. E. Schiesser, G. W. Griffiths, *A Compendium of Partial Differential Equation Models: Method of Lines Analysis with Matlab*, Cambridge University Press, New York, 2009.

[56] Y. Shkolnisky, M. Tygert, V. Rokhlin, *Approximation of bandlimited functions*, Appl. Comput. Harmon. Anal. 21 (3) (2006) 413-420.

[57] D. Slepian, *Prolate spheroidal wave functions, Fourier analysis, and uncertainty - IV: extensions to many dimensions; generalized prolate spheroidal wave functions*, Bell Syst. Tech. J. 43 (1964) 3009-3057.

[58] D. Slepian, *Prolate spheroidal wave functions, Fourier analysis, and uncertainty - V: the discrete case*, Bell Syst. Tech. J. 57 (1978) 1371-1430.

[59] D. Slepian, H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis, and uncertainty - I*, Bell Syst. Tech. J. 40 (1961) 43-63.

[60] A. Solomonoff, *A fast algorithm for spectral differentiation*, J. Comput. Phys. 98 (1992) 174-177.

[61] H. Tal-ezer, *Spectral methods in time for hyperbolic equations*, SIAM J. Numer. Anal. 23 (1) (1996), 11-26.

[62] J. W. Thomas, *Numerical Partial Differential Equations: Finite Difference Methods*, Springer, New York, 2010.

[63] L. N. Trefethen, *Spectral Methods in Matlab*, SIAM, Philadelphia, 2000.

[64] L. N. Trefethen, M. R. Trummer, *An instability phenomenon in spectral methods*, SIAM J. Numer. Anal. 24 (1987) 1008-1023.

[65] H. Vandeven, *On the eigenvalues of second-order spectral differentiation operators*, Comput. Meth. Appl. Mech. Eng. 80 (1-3) (1990) 313-318.

[66] L.-L. Wang *Analysis of spectral approximations using prolate spheroidal wave functions*, Math. Comp. 79 (270) (2010) 807-827.

[67] J. A. C. Weideman, *The eigenvalues of Hermite and rational spectral differentiation matrices*, Numer. Math. 61 (1992) 409-431.

[68] J. A. C. Weideman, L. N. Trefethen, *The eigenvalues of second-order spectral differentiation matrix*, SIAM J. Numer. Anal. 25 (1988) 1279-1928.

[69] H. Xiao, V. Rokhlin, N Yarvin, *Prolate spheroidal wavefunctions, quadrature and interpolation*, Inverse Problems 17 (4) (2001) 805-828.

[70] N. Yarvin, V. Rokhlin, *Generalized Gaussian quadratures and singular value decompositions of integral operators*, SIAM J. Sci. Comput. 20 (1998) 699-718.